(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property
Organization
International Bureau

(43) International Publication Date
17 November 2005 (17.11.2005)

PCT

(10) International Publication Number
WO 2005/109179 A2

(51) International Patent Classification⁷: G06F 7/00

(21) International Application Number:
PCT/US2005/015643

(22) International Filing Date: 5 May 2005 (05.05.2005)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/568,881 6 May 2004 (06.05.2004) US
10/963,306 12 October 2004 (12.10.2004) US

(71) Applicant *(for all designated States except US)*: MACROVISION CORPORATION [US/US]; 2830 De La Cruz Boulevard, Santa Clara, CA 95050 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: BLAND, William [GB/GB]; 21 Guild Court, Redcliff Backs, Bristol BS1 6HX (GB). MOORE, James, Edward [GB/GB]; 3 Orchard Road, Colchester, Essex CO1 1SG (GB).

(74) Agent: SALTER, James, H.; Macrovision Corporation, Patent Department, 2830 De La Cruz Boulevard, Santa Clara, CA 95050 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: IDENTIFICATION OF INPUT FILES USING REFERENCE FILES ASSOCIATED WITH NODES OF A SPARSE BINARY TREE

(57) Abstract: An input profile is generated from an input audio file using a measurable attribute that was also used to generate reference profiles from reference audio files. The input profile is then subjected to a process that was also used to generate a reference profiles tree, which is structured as a sparse binary tree, from the reference profiles. As a result of the process, information of reference profiles having similar characteristics as the input profile, with respect to the measurable attribute, are retrieved from resulting nodes of the reference profiles tree. The input profile is then compared with this subset of the reference profiles, representing potential matches, to determine that either it matches one of the reference profiles, or that it is a spoof, or that it does not match any of the reference profiles.

IDENTIFICATION OF INPUT FILES USING REFERENCE FILES ASSOCIATED WITH NODES OF A SPARSE BINARY TREE

CROSS REFERENCE TO RELATED APPLICATIONS

[0001] This application claims priority to United States Provisional Application Serial No. 60/568,881 filed May 6, 2004, which is incorporated herein by reference; and is a continuation-in-part of commonly-owned United States Application Serial No. 10/472,458, filed September 19, 2003, entitled "Method and Apparatus for Identifying Electronic Files," which is also incorporated herein by reference.

FIELD OF THE INVENTION

[0002] The present invention generally relates to techniques for identifying digitized samples of time varying signals and in particular, to a method and apparatus for identifying input files using reference files associated with nodes of a sparse binary tree.

BACKGROUND OF THE INVENTION

[0003] In searching for particular audio files on the Internet, it is useful to be able to determine the identity of untitled audio files as well as to confirm that titled audio files are what they purport to be. Although a human may conceivably make such determinations and confirmations by simply listening to the content of the audio files by playing them through a media player, such an approach is not always reliable. Also, a process such as this involving human judgment is inherently very slow.

[0004] Therefore, it is advantageous to employ a computer to determine the identity of untitled audio files as well as to confirm that titled audio files are what they purport to be. The computer can not only store a lot of information to assist in identifying an input audio file, it can also process that information very quickly.

[0005] In one technique employing a computer, an algorithm is used to uniquely identify audio file content. Using this approach, a master code is generated by performing the algorithm on content in a master audio file. By applying the same algorithm to the content of an input audio file, the calculated code may then be compared with the master code to determine a match.

[0006] Use of such an algorithm, however, does not always lead to proper identification, because the content of an audio file may not have exactly the same length of recording as the content of the master audio file, for example, by starting at a point a little later in time, thus giving rise to a calculated code that would not match the master code. Also, if the content of the input audio file contains noise spikes or background noise, this would also give rise to a calculated code that would not match the master code. Thus, in both of these cases, the stored content is not properly identified.

OBJECTS AND SUMMARY OF THE INVENTION

[0007] Accordingly, one object of the present invention is to provide a method and apparatus for identifying input files that are reliable even if their content is offset in time, or contains noise spikes or background noise.

[0008] Another object is to provide a method and apparatus for identifying input files that are computationally fast when performed in a computer system.

[0009] Another object is to provide a method and apparatus for identifying input files that minimize data storage requirements in a computer system.

[0010] These and other objects are accomplished by the various aspects of the present invention, wherein briefly stated, one aspect is a method for matching an input audio file with reference audio files, comprising: identifying potential matches of an input audio file among reference audio files based upon at least one common characteristic; and searching for a match of the input audio file among the potential matches.

[0011] Another aspect is a method for matching an input audio file with reference audio files, comprising: identifying potential matches of an input audio file among reference audio files based upon at least one common characteristic; and comparing an input profile resulting from a measurable attribute of the input audio file with reference profiles resulting from the same measurable attribute of the potential matches to determine a match.

[0012] Another aspect is a method for matching an input file with reference files, comprising: identifying potential matches of an input file among reference files by associating nodes of a sparse binary tree with the input file in a same manner used to associate nodes of the sparse binary tree with the reference files; and searching for a match of the input file among the potential matches..

[0013] Another aspect is a method for matching an input file with reference files, comprising: identifying potential matches of an input file among reference files by associating nodes of a sparse binary tree with the input file in a same manner used to associate nodes of the sparse binary tree with the reference files; and comparing a profile resulting from a measurable attribute of the input file with profiles resulting from the same measurable attribute of the potential matches to determine a match.

[0014] Another aspect is a method for matching an input audio file with reference audio files, comprising: generating an input profile from an input audio file based upon a measurable attribute also used to generate reference profiles from reference audio files; identifying potential matches among the reference profiles with the input profile by processing the input profile in a manner used to associate individual of the reference profiles with nodes of a sparse binary tree; and comparing the input profile with the potential matches to determine a match.

[0015] Still another aspect is a method for matching an input audio file with reference audio files, comprising: generating reference profiles from reference audio files using a measurable attribute; generating a sparse binary tree by applying a process to the reference profiles such that identifications of the reference profiles are associated at resulting nodes of the sparse binary tree; generating an input profile from the input audio file using the measurable attribute; applying the process to the input profile so that associated reference profiles are identified from resulting nodes of the sparse binary tree; and comparing at least a portion of the input profile with corresponding portions of the identified reference profiles to determine a match.

[0016] Another aspect is an apparatus for matching an input audio file with reference audio files, comprising at least one computer configured to: identify potential matches of an input audio file among reference audio files based upon at least one common characteristic; and search for a match of the input audio file among the potential matches.

[0017] Another aspect is an apparatus for matching an input audio file with reference audio files, comprising at least one computer configured to: identify potential matches of an input audio file among reference audio files based upon at least one common characteristic; and compare an input profile resulting from a measurable attribute of the input audio file with reference profiles resulting from the same measurable attribute of the potential matches to determine a match.

[0018] Another aspect is an apparatus for matching an input file with reference files, comprising at least one computer configured to: identify potential matches of an input file among reference files by associating nodes of a sparse binary tree with the input file in a same manner used to associate nodes of the sparse binary tree with the reference files; and search for a match of the input file among the potential matches.

[0019] Another aspect is an apparatus for matching an input file with reference files, comprising at least one computer configured to: identify potential matches of an input file among reference files by associating nodes of a sparse binary tree with the input file in a same manner used to associate nodes of the sparse binary tree with the reference files; and compare a profile resulting from a measurable attribute of the input file with profiles resulting from the same measurable attribute of the potential matches to determine a match.

[0020] Another aspect is an apparatus for matching an input audio file with reference audio files, comprising at least one computer configured to: generate an input profile from an input audio file based upon a measurable attribute also used to generate reference profiles from reference audio files; identify potential matches among the reference profiles with the input profile by processing the input profile in a manner used to associate individual of the reference profiles with nodes of a sparse binary tree; and compare the input profile with the potential matches to determine a match.

[0021] Yet another aspect is an apparatus for matching an input audio file with reference audio files, comprising at least one computer configured to: generate reference profiles from reference audio files using a measurable attribute; generate a sparse binary tree by applying a process to the reference profiles such that identifications of the reference profiles are associated at resulting nodes of the sparse binary tree; generate an input profile from the input audio file using the measurable attribute; apply the process to the input profile so that associated reference profiles are identified from resulting nodes of the sparse binary tree; and compare at least a portion of the input profile with corresponding portions of the identified reference profiles to determine a match.

[0022] Additional objects, features and advantages of the various aspects of the present invention will become apparent from the following description of its preferred embodiment, which description should be taken in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

[0023] **FIG. 1** illustrates a data flow diagram for an audio matcher program, utilizing aspects of the present invention.

[0024] **FIG. 2** illustrates a flow diagram of a method for generating reference profiles and a reference profiles tree, utilizing aspects of the present invention.

[0025] **FIG. 3** illustrates a flow diagram of a method for generating profiles from digitized audio clips, utilizing aspects of the present invention.

[0026] **FIG. 4** illustrates a flow diagram of a method for generating a reference profiles tree, utilizing aspects of the present invention.

[0027] **FIG. 5** illustrates a flow diagram of a method for associating reference profile information with nodes of a reference profiles tree, utilizing aspects of the present invention.

[0028] **FIG. 6** illustrates a flow diagram of a method for storing a profile hook into a reference profiles tree for each chunk offset of a reference profile identified as being a local maximum, utilizing aspects of the present invention.

[0029] **FIG. 7** illustrates a flow diagram of a method for generating an input profile from an input audio clip.

[0030] **FIG. 8** illustrates a diagram for functions performed by an audio matcher, utilizing aspects of the present invention.

[0031] **FIG. 9** illustrates a flow diagram of a method for sampling input profile chunks and determining chunk offsets to be used for searching a reference profiles tree for profile hooks, utilizing aspects of the present invention.

[0032] **FIG. 10** illustrates a flow diagram of a method for searching a reference profiles tree for profile hooks corresponding to an input profile chunk offset, utilizing aspects of the present invention.

[0033] **FIG. 11** illustrates a flow diagram of a method for comparing an input profile against reference profiles corresponding to profile hooks retrieved from a reference profiles tree search, utilizing aspects of the present invention.

[0034] **FIG. 12** illustrates a flow diagram of a method for merging mini-matches, utilizing aspects of the present invention.

[0035] **FIG. 13** illustrates a diagram for sub-functions performed by an audio matcher, utilizing aspects of the present invention.

[0036] **FIG. 14** illustrates a flow diagram of a method for identifying an input file as a spoof file, utilizing aspects of the present invention.

[0037] **FIG. 15** illustrates a flow diagram of a method for identifying a best match for an input file, utilizing aspects of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

[0038] All methods, generators and programs described herein are preferably performed on one or more computers cooperating together such as in a distributed or other processing environment.

[0039] Referring to **FIG. 1**, an audio matcher program **100** matches an input profile **101** of an input audio clip to one of a store of reference profiles **102** of reference audio clips. In addition to the input profile **101** and the reference profiles **102**, a reference profiles tree **103** is also used by the audio matcher **100** to facilitate and expedite the matching process, which in this case, results in one of the following outcomes: a determination that the input profile **101** is a spoof (for example, its corresponding input audio clip is not what it purports to be), or an identification of an acceptable best match for the input profile **101** among the reference profiles **102**, or a determination that no acceptable match has been found for the input profile **101** among the reference profiles **102**.

[0040] The reference audio clips in this case may be published music that is protected by copyright law, and the input audio clips may be audio files either residing on user computers or being transmitted through the Internet using a file sharing network. Formats for the audio clips may be any standard format such as MP3.

[0041] **FIG. 2** illustrates, as an example, a method by which the reference profiles and the reference profiles tree are generated. In particular, a profile generator **202** generates the reference profiles **102** from corresponding reference audio clips **201**. A reference profiles tree generator **203** then generates a references profiles tree **103** from the reference profiles **102**. The format of the reference profiles tree **103** is a sparse binary tree in order to minimize data storage requirements and maximize search performance.

[0042] **FIG. 3** illustrates, as an example, a method performed by the profile generator **202** to generate profiles from digitized audio clips. In the method, a chunk represents a programmable period of time such as 0.1 seconds of the audio clip. In **301**, digitized sample information for a first chunk of the audio clip is serially read, and in **302**, the number of zero crossings in the chunk is counted. A zero crossing occurs each time the sign changes between adjacent samples. In **303**, a determination is made whether the current chunk is a last chunk in the profile. If the determination results in a YES, then the profile generator 202 terminates. On the other hand, if the determination is NO, then in **304**, digitized sample information for a next chunk of the audio clip is serially read, and the process continues by jumping back to **302** and repeating **302~304** until the last chunk in the profile has been processed through **302**.

[0043] The profile generator **202** is used to generate reference profiles **102** from reference audio clips **201** (as shown in **FIG. 2**) and to generate an input profile **101** from an input audio clip **701** (as shown in **FIG. 7**). In addition to counting zero crossings as described in reference to **FIG. 3**, the profile generator 202 may also generate other information such as amplitude ratios between successive chunks to better characterize audio clips in its generated profiles and improve matching accuracy by the audio matcher 100 through the use of such enhanced profiles.

[0044] **FIG. 4** illustrates, as an example, a method performed by the reference profiles tree generator **203** to generate a reference profiles tree **103** by including reference profile information for each of the reference profiles **102** in it. As previously described, the reference profiles tree **103** is generated as a sparse binary tree.

[0045] **FIG. 5** illustrates, as an example, a method performed in function **402** of **FIG. 4** to add reference profile information to the reference profiles tree **103**. As noted in **FIG. 4**, this method will be performed for each of the reference profiles **102**, so that information of all of the reference profiles will be stored in the same reference profiles tree **103**. The reference profiles tree **103** and particularly, its sparse binary tree architecture, will subsequently be used by the audio matcher to significantly speed up the matching process.

[0046] As used herein, the term "chunk offset" means the difference in number of chunks between a current chunk of the reference profile and a first chunk of the reference profile, plus one. Thus, the number of the chunk is equal to the chunk offset in this convention.

[0047] Two programmable parameters are used in the method. The term "velocity" means the number of chunks between local maximums in the reference profile, and the term "acceleration" means the change in velocity divided by the number of chunks over which the change occurs. Initial values for velocity and acceleration are pre-defined prior to performance of the function **402**. As an example, the initial velocity may be set to 1, and the initial acceleration may also be set to 1. The velocity is then modified according to the method. The acceleration, on the other hand, is generally constant at its initial value.

[0048] In **501**, the chunk offset is initialized to be equal to the initial velocity. In **502**, a determination is made whether the zero crossing count for the current chunk is a local maximum. To be considered a local maximum, the zero crossing count for the current chunk must be greater by a programmed threshold value than both the zero crossing count for the chunk right before the current chunk and the zero crossing count for the chunk right after the current chunk. In situations where the current chunk does not have either a chunk right before it (i.e., it is the first chunk in the reference profile) or a chunk right after it (i.e., it is the last chunk in the reference profile), a zero will be assumed for the zero crossing count in those cases.

[0049] If the determination in **502** is YES, then in **503**, a profile hook for this chunk offset is stored in the reference profiles tree **103**. Additional details on **503** are described in reference to **FIG. 6** below.

[0050] On the other hand, if the determination in **502** is NO, then in **504**, the chunk offset is incremented by the velocity.

[0051] In **505**, a determination is then made whether the end of the reference file has been reached. This determination would be YES, if the new chunk offset is greater than the chunk number of the last chunk in the reference profile. Therefore, if the determination in **505** is YES, then the method is done, and another reference profile can be processed as shown in **FIG. 4**.

[0052] On the other hand, if the determination in **505** is NO, then in **506**, the velocity is incremented by the acceleration. By incrementing the velocity in this fashion, chunks will be processed in a more efficient manner. Rather then processing every chunk in a reference profile to see if it is a local maximum, chunks are processed in a quadratically increasing fashion to take advantage of the observation that matches between input profiles and reference profiles usually can be determined early on in the profiles.

[0053] The method then loops back to **502** to process the newly calculated chunk offset, and continues looping through **502~506** until the end of the reference profile is determined in **505**.

[0054] **FIG. 6** illustrates, as an example, a method performed in function **503** of **FIG. 5** to store a profile hook into the reference profiles tree **103** for each chunk offset identified in **502** of **FIG. 5** as having a local maximum zero crossing count. In **601** and **602** respectively, the current node at which processing starts on the sparse binary tree is set to its root node and the chunk at which processing starts is set to the chunk offset being processed at the time.

[0055] In **603**, a determination is made whether the zero crossing count for the current chunk is greater than a programmable constant or threshold value. If the determination in **603** is NO, then in **604**, the current node is changed to a right-branch child node, which is created at that time if it doesn't already exist in the reference profiles tree **103**. On the other hand, if the determination in **603** is YES, then in **605**, the current node is changed to a left-branch child node, which is created at that time if it doesn't already exist in the reference profiles tree **103**.

[0056] In **606**, a determination is then made whether the current chunk is the last chunk in the reference profile. If the determination in **606** is NO, then in **607**, the current chunk is incremented by 1, and the method loops back to **603**, and continues looping through **603~607** until the determination in **606** is YES. When the determination in **606** is YES, then in **608**, the method stores the profile hook in the then current node, and is done. The profile hook in this case includes a profile identification or "ID" and the chunk offset that is being processed at the time in function **503**. The profile ID serves to uniquely identify the content of the reference profile in this case.

[0057] In the following description, it is now assumed that generation of the reference profiles tree **103** is complete so that it contains information of profile hooks for each of the reference profiles **102** at various of its nodes.

[0058] **FIG. 7** illustrates, as an example, a method for generating an input profile **101** that parallels the method used for generating each of the reference profiles **102**.

[0059] **FIG. 8** illustrates three primary functions performed by the audio matcher **100**. In a first function **801**, the audio matcher **100** identifies mini-matches of an input profile **101** in the reference profiles tree **103**. If no mini-matches are found, then this function reports back that the input profile is a no-match. Since the no-match is determined early on in the process, this avoids the necessity to perform subsequent processing to determine a best match or that the input profile is for a spoof.

[0060] Assuming mini-matches have been identified between the input profile and one or more reference profiles, then in a second function **802**, the audio matcher **100** then stores and merges when appropriate the mini-matches for subsequent processing. In a third function **803**, the audio matcher **100** then determines one of the following: an acceptable best match for the input profile; a determination that the input profile is a spoof; or a no-match if the input profile is not determined to be a spoof or if an acceptable best match cannot be found.

[0061] **FIG. 9** illustrates, as an example, a method for performing the mini-match identifying function **801** of **FIG. 8**. The method parallels that of **FIG. 5**, wherein local maximums in a reference profile are identified so that profile hooks can be stored in the reference profiles tree **103**. In this case, however, local maximums in the input profile are identified so that mini-matches may be found in the reference profiles tree **103**. In particular, the identification of zero crossing count local maximums in the input profile as performed in **901**, **902** and **904~906** are performed identically as their counterparts **501**, **502** and **504~506** of **FIG. 5** in identifying zero crossing count local maximums in a reference profile.

[0062] In **903**, however, rather than storing a profile hook in the reference profiles tree for the chunk offset as performed in **503** of **FIG. 5**, the chunk offset is used to search for matches in the reference profiles tree **103**.

[0063] **FIG. 10** illustrates, as an example, a method performed in function **903** of **FIG. 9**. The method is similar to that of **FIG. 6**, wherein movement down the reference profiles tree **103** is performed. In this case, however, zero crossing counts in the input profile are used instead of the zero crossing counts in a reference profile to determine the movement down the reference profiles tree **103**.

[0064] Starting in **1001**, the current node in the reference profiles tree **103** is initially set to the root node, and in **1002**, the current chunk is set to the chunk offset currently being processed.

[0065] In **1003**, a determination is made whether the zero crossing count for the current chunk is greater than a programmable constant. The constant that is to be used here is the same as that used in **603** of **FIG. 6**.

[0066] If the determination in **1003** is NO, then in **1004**, the current node is changed to a right-branch child node. On the other hand, if the determination in **1003** is YES, then in **1005**, the current node is changed to a left-branch child node.

[0067] In **1006**, a determination is then made whether the current chunk is the last chunk in the input profile. If the determination in **1006** is NO, then in **1007**, the current chunk is incremented by 1, and the method loops back to **1003**, and continues looping through **1003~1007** until the determination in **1006** is YES. When the determination in **1006** is YES, then in **1008**, the method matches the input profile against all reference profiles identified in profile hooks stored at the current node of the reference profiles tree **103**.

[0068] **FIG. 11** illustrates, as an example, a method for performing the function **1008** of **FIG. 10**. In **1101**, a determination is first made whether there are any reference profiles identified in reference profile hooks stored at the current node of the reference profiles tree **103**. If the determination in **1101** is NO, then the method has nothing more to do so it ends.

[0069] On the other hand, if the determination in **1101** is YES, then in **1102**, the first N chunks of the input profile are compared with the corresponding first N chunks of a first reference profile identified. In **1103**, a determination is made whether they match. In order for corresponding chunks to match, their zero crossing counts do not have to be exactly equal. As long as the absolute difference between the zero crossing counts is within a programmed tolerance, they may be determined to be a match. Also, it may not be necessary for all of the first N chunks to match, the match determination may be a YES as long as a high enough percentage of the first N chunks match.

[0070] If the determination in **1103** is a YES, then in **1104**, a mini-match at the current offset of the input profile is generated. Generation of the mini-match involves including the information in the following table in the mini-match.

Table 1. Information included in a mini-match.

Parameter	**Description**
$time_1$	The offset into the input profile at which this match begins.
$time_2$	The offset into the input profile at which this match ends.
wt_1	The offset into the input profile at which the reference profile begins.
wt_2	The offset into the input profile at which the reference profile ends.
time matched	The amount of match time represented by this mini-match.
full match	"True" if this is a "full" match.
source	The ID for the reference profile corresponding to this mini-match.
err	The total amount of error in this mini-match.
ID	An identifier for the mini-match.

[0071] On the other hand, if the determination in **1103** is a NO, then in **1105**, a determination is made whether there is another reference profile identified at the current node of the reference profiles tree **103**. If the determination in **1105** is YES, then in **1106**, the first N chunks of the input profile are then compared with those of the next identified reference profile, and the method continues by looping through **1103~1106** until either a match is found or there are no more reference profiles to be compared against the input profile.

[0072] If the determination in **1105** results at any time in a NO, then in **1107**, the method generates a "non-full" mini-match using the best matching one of the reference profiles identified at the current node of the reference profiles tree **103** (i.e., the reference profile whose first N chunks came closest to being determined as a match to the first N chunks of the input profile). As with the "full" mini-match generated in **1104**, the "non-full" mini-match will also be associated to the current offset of the input profile.

[0073] **FIG. 12** illustrates, as an example, a method for performing the mini-match storing and merger function **802** of **FIG. 8**. In **1201**, a first one of the mini-matches generated in the function **801** of **FIG. 8** is input. The mini-match can be either a "full" or "non-full" mini-match. In **1202**, a determination is made whether any mini-matches have already been stored in the audio matcher **100** for subsequent processing. If the determination in **1202** is NO (as it will be for the first mini-match being input for the input profile), then in **1203**, the mini-match is added to the store and the method jumps down to **1207**.

[0074] In **1207**, a determination is then made whether there are any more mini-matches to be input. If the determination in **1207** is YES, then the method jumps back to **1201** to input the next mini-match. In **1202**, a determination is once again made whether there are any stored mini-matches. This time, since the first mini-match was stored, the determination will result in a YES, so that the method proceeds to **1204**.

[0075] In **1204**, a search is performed to find a merger candidate for the current mini-match among the mini-matches already in the store. In order to be considered a merger candidate, the current mini-match and the stored mini-match must refer to the same reference profile ID, and any difference between their respective wt_1 parameters (offsets into the input profile at which the reference profile begins) must be within a specified tolerance such as 50 chunks or 5 seconds.

[0076] In **1205**, a determination is then made whether a merger candidate has been found. If the determination in **1205** is NO, then the current mini-match is added to the store in **1203**, and the method proceeds from there as previously described.

[0077] On the other hand, if the determination in **1205** is YES, then in **1206**, the current mini-match is merged with the merger candidate. When merging the current mini-match with the merger candidate, the parameter values for wt_1, wt_2, $time_1$ and $time_2$ of the merged mini-match are weighted averages of the current mini-match and the merger candidate values, weighted by their respective matched times. The parameter value for "err" of the merged mini-match is the sum of the current mini-match and the merger candidate values. If either the current mini-match or the merger candidate is a "full" match, then the merged mini-match has its full match parameter set to true.

[0078] After merger, the method proceeds to **1207**.

[0079] In **1207**, a determination is made whether there are any more mini-matches to be processed. If the determination in **1207** is YES, then the method proceeds by looping through **1201~1207** until all mini-matches have been processed by either being stored individually in the audio matcher store or merged with another mini-match already stored in the audio matcher store, and the determination in **1207** at that time results in a NO.

[0080] **FIG. 13** illustrates, as an example, a software structure for implementing the function **803** of **FIG. 8** in which a first function **1301** determines if the input profile is a spoof, and a second function **1302** finds a best match from the store of mini-matches generated in **802** of **FIG. 8**. The first and second functions **1301** and **1302** may be performed serially, or in parallel as shown. If both functions **1301** and **1302** fail (i.e., the first function **1301** fails to identify the input profile **101** as a spoof and the second function **1302** fails to find an acceptable best match), then in **1303**, it is determined that the input profile **101** is a no-match (i.e., no match has been found for it among the reference profiles **102**).

[0081] **FIG. 14** illustrates, as an example, a method for performing the first function **1301** to determine whether the input profile is a spoof. In **1401**, a determination is made whether there is more than one reference profile identified by the mini-matches in the store. If the determination in **1401** results in a NO, then in **1402**, a no spoof found conclusion is made and the method stops at that point.

[0082] On the other hand, if the determination in **1401** is YES, then in **1403**, a determination is made whether the sum of the time matched for all the mini-matches in the store is greater than some threshold percentage of the input profile such as, for example, 70%. If the determination in **1403** results in a NO, then in **1402**, a no spoof found conclusion is made and the method stops at that point.

[0083] On the other hand, if the determination in **1403** is YES, then in **1404**, a determination is made whether each mini-match has an error/second value that is less than some maximum value. The error/second value for each mini-match may be calculated by the ratio of the mini-match's "err" parameter and "time matched" parameter. If the determination in **1404** results in a NO, then [illegible] conclusion is made and the method stops at that point.

[0084] On the other hand, if the determination in **1404** is YES, then in **1405**, a spoof found conclusion is made and the method stops at that point. In this case, the spoof may be formed by compositing several tracks together or [illegible] of one track. Since these kinds of spoofs are quite common on peer-to-peer networks, the ability to automatically identify them is useful.

[0085] **FIG. 15** illustrates, as an example, a method for performing the second function **1302** to find an acceptable best match for the input profile **101**. In **1501** and **1502**, the method starts by ignoring all "non-full" mini-matches and all mini-matches having an errors/second greater than a maximum allowable value.

[0086] In **1503**, the method then identifies one of the remaining mini-matches as a best match according to programmed criteria such as its errors/second value, its time matched value, and the percentage of its reference profile that it recognizes. Typically, the best match will be a mini-match that exceeds all other mini-matches in all of these criteria. In the event that two mini-matches are close, some weighting of the criteria may be performed to determine a best match between the two.

[0087] In **1504**, a determination is then made whether the percentage of the input profile and the reference profile covered by the best match exceeds some minimum value. If the determination in **1504** is YES, then in **1505**, the best match identified in **1503** is concluded to be an acceptable best match and the method ends at that point. On the other hand, if the determination in **1504** is NO, then the best match identified in **1503** is concluded in **1506** to be an unacceptable best match and the method ends at that point with a conclusion in this case that no acceptable best match was found.

[0088] Although the various aspects of the present invention have been described with respect to a preferred embodiment, it will be understood that the invention is entitled to full protection within the full scope of the appended claims.

CLAIMS

We claim:

1. A method for matching an input audio file with a plurality of reference audio files, comprising:

identifying potential matches of an input audio file among a plurality of reference audio files based upon at least one common characteristic; and

searching for a match of the input audio file among the potential matches.

2. The method according to claim 1, wherein the identification of the potential matches comprises identifying potential matches of the input audio file among the plurality of reference audio files by associating nodes of a sparse binary tree with the input audio file in a same manner used to associate nodes of the sparse binary tree with the plurality of reference audio files.

3. The method according to claim 1, wherein the identification of potential matches comprises:

generating an input profile from the input audio file based upon a measurable attribute also used to generate a plurality of reference profiles from the plurality of reference audio files;

identifying potential matches among the plurality of reference profiles with the input profile by processing the input profile in a manner used to associate the plurality of reference profiles with nodes of a sparse binary tree.

4. The method according to claim 3, wherein the generation of the input profile comprises:

segmenting the input audio file into chunks representing a period of time; and counting a number of occurrences of a measurable attribute of the input audio file in individual of the chunks.

5. The method according to claim 4, wherein individual of the chunks includes information of digitized samples of an audio clip over the period of time, and the measurable attribute for individual of the chunks is a number of zero crossings of the digitized samples in the chunk.

6. The method according to claim 4, wherein individual of the plurality of reference profiles has been generated from a corresponding one of the plurality of reference audio files by segmenting the reference audio file into chunks representing the period of time, and counting a number of occurrences of the measurable attribute of the reference audio file in individual of the chunks.

7. The method according to claim 6, wherein the plurality of reference profiles are associated with nodes of the sparse binary tree by sampling chunks in individual of the reference profiles so that if a sampled chunk has a measurable attribute satisfying a criterion, then a profile hook including an identification of the reference profile is associated with a node in the sparse binary tree.

8. The method according to claim 7, wherein individual of the chunks of the input audio file includes information of digitized samples of an audio clip over their corresponding period of time, and the measurable attribute for individual of the chunks is a number of zero crossings of the digitized samples in the chunk.

9. The method according to claim 8, wherein the criterion is satisfied if the zero crossing count of a sampled chunk is a local maximum.

10. The method according to claim 7, wherein the profile hook is associated with a node in the sparse binary tree by storing the profile hook at a current node resulting from moving down left or right branch child nodes starting with a root node of the sparse binary tree depending upon whether a current chunk has a measured attribute value greater than a specified value starting with

the sampled chunk having the measurable attribute that satisfies the criterion and ending with a last chunk of the reference profile.

11. The method according to claim 7, wherein the sampling of chunks in individual of the reference profiles is performed on an ever increasing sampling basis.

12. The method according to claim 11, wherein the ever increasing sampling basis is a quadratically increasing sample basis.

13. The method according to claim 11, wherein the ever increasing sampling basis is an exponentially increasing sample basis.

14. The method according to claim 11, wherein the sampling of chunks in individual of the reference profiles is performed on an ever increasing sampling basis starting with the sampled chunk having the measurable attribute that satisfies the criterion and ending with the last chunk of the reference profile by incrementing through the chunks at a specified velocity and acceleration.

15. The method according to claim 14, wherein the velocity is the number of chunks between local maxima in the reference profile, and the acceleration is the change in velocity divided by the number of chunks over which the change occurs.

16. The method according to claim 10, wherein the identification of potential matches among the plurality of reference profiles comprises identifying potential mini-matches by retrieving profile hooks associated with nodes in the sparse binary tree by identifying profile hooks associated with a current node resulting from moving down left or right branch child nodes starting with the root node of the sparse binary tree depending upon whether a current chunk of the input profile has a measured attribute value greater than the specified value starting with the sampled chunk of the input profile having the measurable attribute that satisfies the criterion and ending with the last chunk of the input profile.

17. The method according to claim 16, wherein the identification of mini-matches among the plurality of reference profiles further comprises for individual reference profiles corresponding to the retrieved profile hooks:

comparing a number of chunks of the input profile and corresponding chunks of the reference profile; and

identifying a mini-match if corresponding chunks of the reference profile substantially matches those of the input profile.

18. The method according to claim 17, wherein the identification of mini-matches among the plurality of reference profiles further comprises: identifying a non-full mini-match using a best matching one of the reference profiles with the input profile if none of the reference profiles identified by the profile hooks substantially matches those of the input profile.

19. The method according to claim 18, wherein the identification of potential matches further comprises: merging any mini-matches and non-full mini-matches corresponding to the same reference profile and having an offset into the input profile at which the reference profile begins within a specified tolerance.

20. The method according to claim 19, wherein if a mini-match is merged with a non-full mini-match, then the merged entity is referred to as a mini-match.

21. The method according to claim 20, further comprising: identifying the input audio file as a spoof if all mini-matches identified for the input profile do not refer to the same reference profile.

22. The method according to claim 21, wherein the input audio file is not identified as a spoof if the sum of the total audio time covered by the mini-matches is less than a specified threshold value.

23. The method according to claim 22, wherein the input audio file is not identified as a spoof even if the sum of the total audio time covered by

the mini-matches is not less than the specified threshold value, if any of the mini-matches has an associated error per second value that is greater than a first specified maximum value.

24. The method according to claim 25, wherein the searching for the match results in a best match being found if a percentage of the input profile and the reference profile covered by the best match exceeds some minimum value after ignoring all non-full mini-matches, ignoring mini-matches having an error per second value that is greater than a second specified maximum value, and taking into consideration any other programmed criteria.

25. A method for matching an input audio file with a plurality of reference audio files, comprising:

identifying potential matches of an input audio file among a plurality of reference audio files based upon at least one common characteristic; and

comparing an input profile resulting from a measurable attribute of the input audio file with reference profiles resulting from the same measurable attribute of the potential matches to determine a match.

26. The method according to claim 25, further comprising generating the input profile by:

segmenting the input audio file into chunks representing a period of time; and

counting a number of occurrences of a measurable attribute of the input audio file in individual of the chunks.

27. The method according to claim 26, wherein individual of the chunks includes information of digitized samples of an audio clip over the period of time, and the measurable attribute for individual of the chunks is a number of zero crossings of the digitized samples in the chunk.

28. The method according to claim 26, wherein individual of the plurality of reference profiles has been generated from a corresponding one of

the plurality of reference audio files by segmenting the reference audio file into chunks representing the period of time, and counting a number of occurrences of the measurable attribute of the reference audio file in individual of the chunks.

29. The method according to claim 28, wherein the plurality of reference profiles are associated with nodes of the sparse binary tree by sampling chunks in individual of the reference profiles so that if a sampled chunk has a measurable attribute satisfying a criterion, then a profile hook including an identification of the reference profile is associated with a node in the sparse binary tree.

30. The method according to claim 29, wherein individual of the chunks of the input audio file includes information of digitized samples of an audio clip over their corresponding period of time, and the measurable attribute for individual of the chunks is a number of zero crossings of the digitized samples in the chunk.

31. The method according to claim 30, wherein the criterion is satisfied if the zero crossing count of a sampled chunk is a local maximum.

32. The method according to claim 29, wherein the profile hook is associated with a node in the sparse binary tree by storing the profile hook at a current node resulting from moving down left or right branch child nodes starting with a root node of the sparse binary tree depending upon whether a current chunk has a measured attribute value greater than a specified value starting with the sampled chunk having the measurable attribute that satisfies the criterion and ending with a last chunk of the reference profile.

33. The method according to claim 29, wherein the sampling of chunks in individual of the reference profiles is performed on an ever increasing sampling basis.

34. The method according to claim 32, wherein the identification of potential matches among the plurality of reference profiles comprises identifying potential mini-matches by retrieving profile hooks

associated with nodes in the sparse binary tree by identifying profile hooks associated with a current node resulting from moving down left or right branch child nodes starting with the root node of the sparse binary tree depending upon whether a current chunk of the input profile has a measured attribute value greater than the specified value starting with the sampled chunk of the input profile having the measurable attribute that satisfies the criterion and ending with the last chunk of the input profile.

35. The method according to claim 34, wherein the comparison [illegible] with the reference profiles of the potential matches comprises for individual reference profiles corresponding to the retrieved profile hooks:

comparing a number of chunks of the input profile and corresponding chunks of the reference profile; and

identifying a mini-match if corresponding chunks of the reference profile substantially matches those of the input profile.

36. The method according to claim 35, wherein the comparison of the input profile with the reference profiles of the potential matches further comprises: identifying a non-full mini-match using a best matching one of the reference profiles with the input profile if none of the reference profiles identified by the profile hooks substantially matches those of the input profile.

37. The method according to claim 36, wherein the comparison of the input profile with the reference profiles of the potential matches further comprises: merging any mini-matches and non-full mini-matches corresponding to the same reference profile and having an offset into the input profile at which the reference profile begins within a specified tolerance.

38. The method according to claim 37, wherein if a mini-match is merged with a non-full mini-match, then the merged entity is referred to as a mini-match.

39. The method according to claim 38, further comprising: identifying the input audio file as a spoof if all mini-matches identified for the input profile do not refer to the same reference profile.

40. The method according to claim 38, wherein the comparison of the input profile with the reference profiles of the potential matches results in a best match being found if a percentage of the input profile and the reference profile covered by the best match exceeds some minimum value after ignoring all non-full mini-matches, ignoring mini-matches having an error per second value [illegible] than a second specified maximum value, and taking into consideration any other programmed criteria.

41. A method for matching an input file with a plurality of reference files, comprising:

identifying potential matches of an input file among a plurality of reference files by associating nodes of a sparse binary tree with the input file in a same manner used to associate nodes of the sparse binary tree with the plurality of reference files; and

searching for a match of the input file among the potential matches.

42. The method according to claim 41, wherein the input file and the reference files include digitized samples of time varying signals.

43. The method according to claim 42, wherein the identification of potential matches comprises:

generating an input profile from the input file based upon a measurable attribute also used to generate a plurality of reference profiles from the plurality of reference files;

identifying potential matches among the plurality of reference profiles with the input profile by processing the input profile in a manner used to associate the plurality of reference profiles with nodes of the sparse binary tree.

44. The method according to claim 43, wherein the generation of the input profile comprises:

segmenting the input file into chunks representing a period of time; and

counting a number of occurrences of a measurable attribute of the input file in individual of the chunks.

45. The method according to claim 44, wherein individual of the chunks includes information of digitized samples of an audio clip over the period of time, and the measurable attribute for individual of the chunks is a number of zero crossings of the digitized samples in the chunk.

46. The method according to claim 45, wherein individual of the plurality of reference profiles has been generated from a corresponding one of the plurality of reference files by segmenting the reference file into chunks representing the period of time, and counting a number of occurrences of the measurable attribute of the reference file in individual of the chunks.

47. The method according to claim 46, wherein the plurality of reference profiles are associated with nodes of the sparse binary tree by sampling chunks in individual of the reference profiles so that if a sampled chunk has a measurable attribute satisfying a criterion, then a profile hook including an identification of the reference profile is associated with a node in the sparse binary tree.

48. The method according to claim 47, wherein individual of the chunks of the input file includes information of digitized samples of an audio clip over their corresponding period of time, and the measurable attribute for individual of the chunks is a number of zero crossings of the digitized samples in the chunk.

49. The method according to claim 47, wherein the profile hook is associated with a node in the sparse binary tree by storing the profile hook at a current node resulting from moving down left or right branch child

nodes starting with a root node of the sparse binary tree depending upon whether a current chunk has a measured attribute value greater than a specified value starting with the sampled chunk having the measurable attribute that satisfies the criterion and ending with a last chunk of the reference profile.

50. The method according to claim 47, wherein the sampling of chunks in individual of the reference profiles is performed on an ever increasing sampling basis.

51. The method according to claim [illegible], wherein the [illegible] [illegible] potential matches among the plurality of reference profiles comprises identifying potential mini-matches by retrieving profile hooks associated with nodes in the sparse [illegible] by [illegible] profile hooks associated with [illegible] resulting from [illegible] [illegible] or right branch [illegible] with the root node of the sparse binary tree depending upon whether a current chunk of the input profile has a measured attribute value greater than the specified value starting with the sampled chunk of the input profile having the measurable attribute that satisfies the criterion and ending with the last chunk of the input profile.

52. The method according to claim 51, wherein the identification of mini-matches among the plurality of reference profiles further comprises for individual reference profiles corresponding to the retrieved profile hooks:

comparing a number of chunks of the input profile and corresponding chunks of the reference profile; and

identifying a mini-match if corresponding chunks of the reference profile substantially matches those of the input profile.

53. The method according to claim 52, wherein the identification of mini-matches among the plurality of reference profiles further comprises: identifying a non-full mini-match using a best matching one of the

reference profiles with the input profile if none of the reference profiles identified by the profile hooks substantially matches those of the input profile.

54. The method according to claim 53, wherein the identification of potential matches further comprises: merging any mini-matches and non-full mini-matches corresponding to the same reference profile and having an offset into the input profile at which the reference profile begins within a specified tolerance.

55. The method according to claim 54, wherein if a mini-match is merged with a non-full mini-match, then the merged entity is referred to as a mini-match.

56. The method according to claim 55, further comprising: identifying the input file as a spoof if all mini-matches identified for the input profile do not refer to the same reference profile.

57. The method according to claim 55, wherein the searching [illegible] match results in a best match being found if a percentage of the input profile and the reference profile covered by the best match exceeds some minimum value after ignoring all non-full mini-matches, ignoring mini-matches having an error per second value that is greater than a second specified maximum value, and taking into consideration any other programmed criteria.

58. A method for matching an input file with a plurality of reference files, comprising:

identifying potential matches of an input file among a plurality of reference files by associating nodes of a sparse binary tree with the input file in a same manner used to associate nodes of the sparse binary tree with the plurality of reference files; and

comparing a profile resulting from a measurable attribute of the input file with profiles resulting from the same measurable attribute of the potential matches to determine a match.

59. The method according to claim 1c, wherein the input file and the reference files include digitized samples of time varying signals.

60. The method according to claim 59, further comprising generating the input profile by:

segmenting the input file into chunks representing a period of time; and

counting a number of occurrences of a measurable attribute of the input [illegible] individual of the chunks.

61. The method according to claim 60, wherein individual of the chunks includes information of [illegible] over the period of [illegible] measurable attribute for individual [illegible] chunks is a number of zero crossings of the [illegible] samples in the chunk.

62. The method according [illegible] wherein individual of the plurality of reference profiles has been generated from a corresponding one of the plurality of reference files by segmenting the reference file into chunks representing the period of time, and counting a number of occurrences of the measurable attribute of the reference file in individual of the chunks.

63. The method according to claim 62, wherein the plurality of reference profiles are associated with nodes of the sparse binary tree by sampling chunks in individual of the reference profiles so that if a sampled chunk has a measurable attribute satisfying a criterion, then a profile hook including an identification of the reference profile is associated with a node in the sparse binary tree.

64. The method according to claim 63, wherein individual of the chunks of the input file includes information of digitized samples of an audio clip over their corresponding period of time, and the measurable attribute for individual of the chunks is a number of zero crossings of the digitized samples in the chunk.

65. The method according to claim 64, wherein the criterion is satisfied if the zero crossing count of a sampled chunk is a local maximum.

66. The method according to claim 63, wherein the profile hook is associated with a node in the sparse binary tree by storing the profile hook at a current node resulting from moving down left or right branch child nodes starting with a root node of the sparse binary tree depending upon whether a current chunk has a measured attribute value greater than a specified value starting with the sampled chunk having the measurable attribute that satisfies the criterion and ending with a last chunk of the reference profile.

67. The method according to claim 63, wherein the sampling of chunks in individual of the reference profiles is performed on an ever increasing sampling basis.

68. The method according to claim 66, wherein the identification of potential matches among the plurality of reference profiles comprises identifying potential mini-matches by retrieving profile hooks associated with nodes in the sparse binary tree by identifying profile hooks associated with a current node resulting from moving down left or right branch child nodes starting with the root node of the sparse binary tree depending upon whether a current chunk of the input profile has a measured attribute value greater than the specified value starting with the sampled chunk of the input profile having the measurable attribute that satisfies the criterion and ending with the last chunk of the input profile.

69. The method according to claim 68, wherein the comparison of the input profile with the reference profiles of the potential matches comprises for individual reference profiles corresponding to the retrieved profile hooks:

comparing a number of chunks of the input profile and corresponding chunks of the reference profile; and

identifying a mini-match if corresponding chunks of the reference profile substantially matches those of the input profile.

70. The method according to claim 69, wherein the comparison of the input profile with the reference profiles of the potential matches further comprises: merging any mini-matches corresponding to the same reference profile and having an offset into the input profile at which the reference profile begins within a specified tolerance.

71. The method according to claim 70, wherein the comparison of the input profile with the reference profiles of the potential matches results in a best match being found if a percentage of the input profile and the reference profile covered by the best match exceeds some minimum value after ignoring mini-matches having an error per second value that is greater than a specified maximum value, and taking into consideration any programmed criteria.

72. A method for matching an input audio file with a plurality of reference audio files, comprising:

generating an input profile from an input audio file based upon a measurable attribute also used to generate a plurality of reference profiles from a plurality of reference audio files;

identifying potential matches among the plurality of reference profiles with the input profile by processing the input profile in a manner used to associate individual of the plurality of reference profiles with nodes of a sparse binary tree; and

comparing the input profile with the potential matches to determine a match.

73. The method according to claim 72, wherein the generation of the input profile comprises:

segmenting the input audio file into chunks representing a period of time; and

counting a number of occurrences of a measurable attribute of the input audio file in individual of the chunks.

74. The method according to claim 73, wherein individual of the chunks includes information of digitized samples of an audio clip over the period of time, and the measurable attribute for individual of the chunks is a number of zero crossings of the digitized samples in the chunk.

75. The method according to claim 73, wherein individual of the plurality of reference profiles has been generated from a corresponding one of the plurality of reference audio files by segmenting the reference audio file into chunks representing the period of time, and counting a number of occurrences of the measurable attribute of the reference audio file in individual of the chunks.

76. The method according to claim 75, wherein the plurality of reference profiles are associated with nodes of the sparse binary tree by sampling chunks in individual of the reference profiles so that if a sampled chunk has a measurable attribute satisfying a criterion, then a profile hook including an identification of the reference profile is associated with a node in the sparse binary tree.

77. The method according to claim 76, wherein individual of the chunks of the input audio file includes information of digitized samples of an audio clip over their corresponding period of time, and the measurable attribute for individual of the chunks is a number of zero crossings of the digitized samples in the chunk.

78. The method according to claim 76, wherein the profile hook is associated with a node in the sparse binary tree by storing the profile hook at a current node resulting from moving down left or right branch child nodes starting with a root node of the sparse binary tree depending upon whether a current chunk has a measured attribute value greater than a specified value starting with the sampled chunk having the measurable attribute that satisfies the criterion and ending with a last chunk of the reference profile.

79. The method according to claim 78, wherein the identification of potential matches among the plurality of reference profiles

comprises identifying potential mini-matches by retrieving profile hooks associated with nodes in the sparse binary tree by identifying profile hooks associated with a current node resulting from moving down left or right branch child nodes starting with the root node of the sparse binary tree depending upon whether a current chunk of the input profile has a measured attribute value greater than the specified value starting with the sampled chunk of the input profile having the measurable attribute that satisfies the criterion and ending with the last chunk of the input profile.

80. The method according to claim 79, wherein the identification of mini-matches among the plurality of reference profiles further comprises for individual reference profiles corresponding to the retrieved profile hooks:

comparing a number of chunks of the input profile and corresponding chunks of the reference profile; and

identifying a mini-match if corresponding chunks of the reference profile substantially matches those of the input profile.

81. The method according to claim 79, wherein the identification of potential matches further comprises: merging any mini-matches corresponding to the same reference profile and having an offset into the input profile at which the reference profile begins within a specified tolerance.

82. The method according to claim 81, further comprising: identifying the input audio file as a spoof if all mini-matches identified for the input profile do not refer to the same reference profile.

83. The method according to claim 81, wherein the searching for the match results in a best match being found if a percentage of the input profile and the reference profile covered by the best match exceeds some minimum value after ignoring mini-matches having an error per second value that is greater than a specified maximum value, and taking into consideration any programmed criteria.

84. A method for matching an input audio file with a plurality of reference audio files, comprising:

generating a plurality of reference profiles from a plurality of reference audio files using a measurable attribute;

generating a sparse binary tree by applying a process to the plurality of reference profiles such that identifications of the plurality of reference profiles are associated at resulting nodes of the sparse binary tree;

generating an input profile from the input audio file using the measurable attribute;

applying the process to the input profile so that associated reference profiles are identified from resulting nodes of the sparse binary tree; and

comparing at least a portion of the input profile with corresponding portions of the identified reference profiles to determine a match.

[illegible] according to claim 84, wherein the generation of the plurality of reference profiles comprises:

segmenting individual of the plurality of reference audio files into chunks individually representing a period of time; and

counting a number of occurrences of the measurable attribute in individual of the chunks of the plurality of reference audio files.

86. The method according to claim 85, wherein individual of the chunks of the reference audio files includes information of digitized samples of an audio clip over the period of time, and the measurable attribute for individual of the chunks is a number of zero crossings of the digitized samples in the chunk.

87. The method according to claim 84, wherein the generation of the sparse binary tree comprises: sampling chunks in individual of the plurality of reference profiles so that if a sampled chunk has a measurable attribute

satisfying a criterion, then a profile hook including an identification of the reference profile is associated with a node in the sparse binary tree.

88. The method according to claim 84, wherein the generation of the input profile comprises:

segmenting the input audio file into chunks representing a period of time; and

counting a number of occurrences of a measurable attribute of the input audio file in individual of the chunks.

89. The method according to claim 88, wherein individual of the chunks of the input audio file includes information of digitized samples of an audio clip over their corresponding period of time, and the measurable attribute for individual of the chunks is a number of zero crossings of the digitized samples in the chunk.

90. The method according to claim 89, wherein the criterion is satisfied if the zero crossing count of a sampled chunk is a local maximum.

91. The method according to claim 89, wherein the profile hook is associated with a node in the sparse binary tree by storing the profile hook at a current node resulting from moving down left or right branch child nodes starting with a root node of the sparse binary tree depending upon whether a current chunk has a measured attribute value greater than a specified value starting with the sampled chunk having the measurable attribute that satisfies the criterion and ending with a last chunk of the reference profile.

92. The method according to claim 84, wherein the application of the process to the input file comprises: identifying potential mini-matches by retrieving profile hooks associated with nodes in the sparse binary tree by identifying profile hooks associated with a current node resulting from moving down left or right branch child nodes starting with the root node of the sparse binary tree depending upon whether a current chunk of the input profile has a measured attribute value greater than the specified value starting with the

sampled chunk of the input profile having the measurable attribute that satisfies the criterion and ending with the last chunk of the input profile.

93. The method according to claim 84, wherein the comparison of the input profile with the identified reference profiles comprises:

comparing a number of chunks of the input profile and corresponding chunks of the reference profiles; and

identifying a mini-match if any corresponding chunks of the reference profiles substantially matches those of the input profile.

94. The method according to claim 93, wherein the comparison of the input profile with the identified reference profiles further comprises: merging any mini-matches corresponding to the same reference profile and having an offset into the input profile at which the reference profile begins within a specified tolerance.

95. The method according to claim 94, further comprising: identifying the input audio file as a spoof if all mini-matches identified for the input profile do not refer to the same reference profile.

96. The method according to claim 95, wherein the comparison of the input profile with the identified reference profiles results in a best match being found if a percentage of the input profile and the reference profile corresponding to the best match exceeds some minimum value after ignoring mini-matches having an error per second value that is greater than a specified maximum value, and taking into consideration any programmed criteria.

97. An apparatus for matching an input audio file with a plurality of reference audio files, comprising at least one computer configured to:

identify potential matches of an input audio file among a plurality of reference audio files based upon at least one common characteristic; and

search for a match of the input audio file among the potential matches.

98. The apparatus according to claim 97, wherein the at least one computer is further configured to identify potential matches of the input audio file among the plurality of reference audio files by associating nodes of a sparse binary tree with the input audio file in a same manner used to associate nodes of the sparse binary tree with the plurality of reference audio files.

99. An apparatus for matching an input audio file with a plurality of reference audio files, comprising at least one computer configured to:

identify potential matches of an input audio file among a plurality of reference audio files based upon at least one common characteristic; and

compare an input profile resulting from a measurable attribute of the input audio file with reference profiles resulting from the same measurable attribute of the potential matches to determine a match.

100. The apparatus according to claim 99, wherein the at least one computer is further configured to generate the input profile by:

segmenting the input audio file into chunks representing a period of time; and

counting a number of occurrences of a measurable attribute of the input audio file in individual of the chunks.

101. The apparatus according to claim 100, wherein individual of the plurality of reference profiles has been generated from a corresponding one of the plurality of reference audio files by segmenting the reference audio file into chunks representing the period of time, and counting a number of occurrences of the measurable attribute of the reference audio file in individual of the chunks.

102. The apparatus according to claim 101, wherein the plurality of reference profiles are associated with nodes of a sparse binary tree by sampling chunks in individual of the reference profiles so that if a sampled chunk has a measurable attribute satisfying a criterion, then a profile hook including an identification of the reference profile is associated with a node in the sparse binary tree.

103. The apparatus according to claim 102, wherein the profile hook is associated with a node in the sparse binary tree by storing the profile hook at a current node resulting from moving down left or right branch child nodes starting with a root node of the sparse binary tree depending upon whether a current chunk has a measured attribute value greater than a specified value starting with the sampled chunk having the measurable attribute that satisfies the criterion and ending with a last chunk of the reference profile.

104. The apparatus according to claim 103, wherein the at least one computer is further configured to identify potential mini-matches by retrieving profile hooks associated with nodes in the sparse binary tree by identifying profile hooks associated with a current node resulting from moving down left or right branch child nodes starting with the root node of the sparse binary tree depending upon whether a current chunk of the input profile has a measured attribute value greater than the specified value starting with the sampled chunk of the input profile having the measurable attribute that satisfies the criterion and ending with the last chunk of the input profile.

105. The apparatus according to claim 104, wherein the at least one computer is further configured to merge any mini-matches corresponding to the same reference profile and having an offset into the input profile at which the reference profile begins within a specified tolerance.

106. The apparatus according to claim 105, wherein the at least one computer is further configured to identify the input audio file as a spoof if all mini-matches identified for the input profile do not refer to the same reference profile.

107. The apparatus according to claim 106, wherein the at least one computer is further configured to not identify the input audio file as a spoof if the sum of the total audio time covered by the mini-matches is less than a specified threshold value.

108. The apparatus according to claim 107, wherein the at least one computer is further configured to not identify the input audio file as a spoof even if the sum of the total audio time covered by the mini-matches is not less than the specified threshold value, if any of the mini-matches has an associated error per second value that is greater than a first specified maximum value.

109. An apparatus for matching an input file with a plurality of reference files, comprising at least one computer configured to:

identify potential matches of an input file among a plurality of reference files by associating nodes of a sparse binary tree with the input file in a same manner used to associate nodes of the sparse binary tree with the plurality of reference files; and

search for a match of the input file among the potential matches.

110. The apparatus according to claim 109, wherein the input file and the reference files include digitized samples of time varying signals.

111. An apparatus for matching an input file with a plurality of reference files, comprising at least one computer configured to:

identify potential matches of an input file among a plurality of reference files by associating nodes of a sparse binary tree with the input file in a same manner used to associate nodes of the sparse binary tree with the plurality of reference files; and

compare a profile resulting from a measurable attribute of the input file with profiles resulting from the same measurable attribute of the potential matches to determine a match.

112. The apparatus according to claim 111, wherein the input file and the reference files include digitized samples of time varying signals.

113. An apparatus for matching an input audio file with a plurality of reference audio files, comprising at least one computer configured to:

generate an input profile from an input audio file based upon a measurable attribute also used to generate a plurality of reference profiles from a plurality of reference audio files;

identify potential matches among the plurality of reference profiles with the input profile by processing the input profile in a manner used to associate individual of the plurality of reference profiles with nodes of a sparse binary tree; and

[illegible] input profile with the potential matches to determine [illegible]

114. An apparatus for matching an input audio file with a plurality [illegible] comprising at least one computer configured to:

generate [illegible] plurality of reference profiles from [illegible] reference audio files using a measurable attribute;

generate a sparse binary tree by applying a process to the plurality of reference profiles such that identifications of the plurality of reference profiles are associated at resulting nodes of the sparse binary tree;

generate an input profile from the input audio file using the measurable attribute;

apply the process to the input profile so that associated reference profiles are identified from resulting nodes of the sparse binary tree; and

compare at least a portion of the input profile with corresponding portions of the identified reference profiles to determine a match.

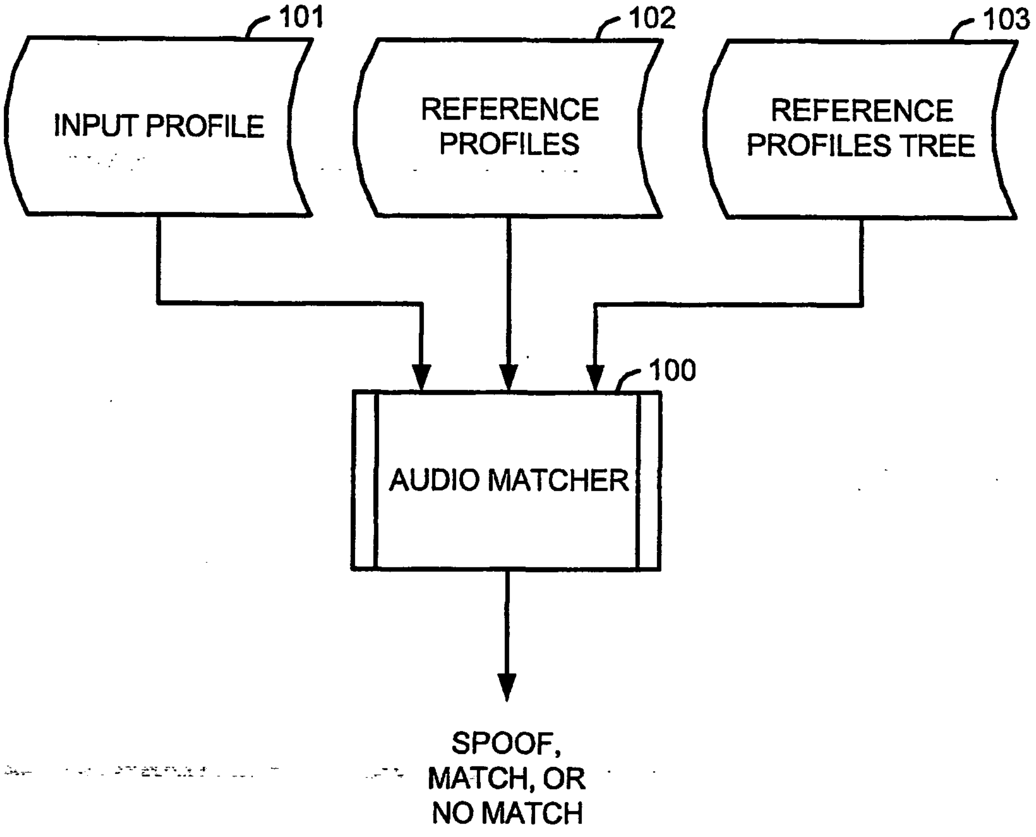
101
INPUT PROFILE
102
REFERENCE PROFILES
103
REFERENCE PROFILES TREE
100
AUDIO MATCHER
SPOOF, MATCH, OR NO MATCH

FIG.1

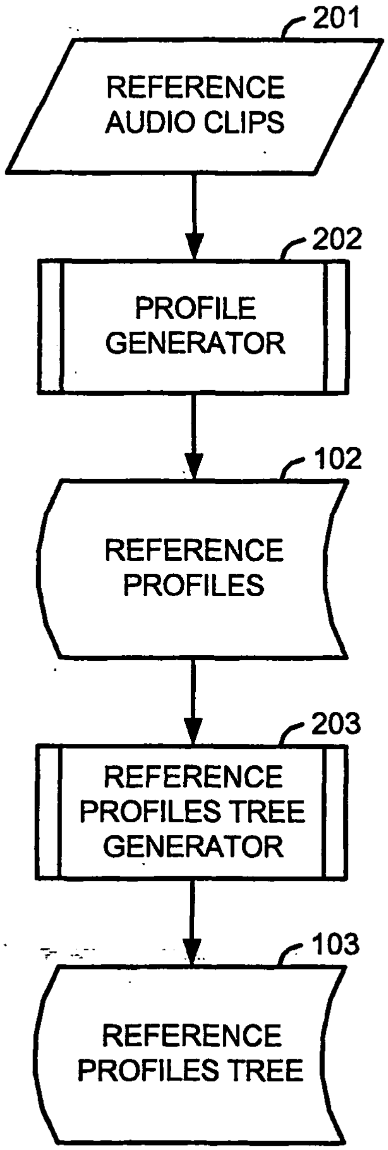
201
REFERENCE AUDIO CLIPS
202
PROFILE GENERATOR
102
REFERENCE PROFILES
203
REFERENCE PROFILES TREE GENERATOR
103
REFERENCE PROFILES TREE

FIG.2

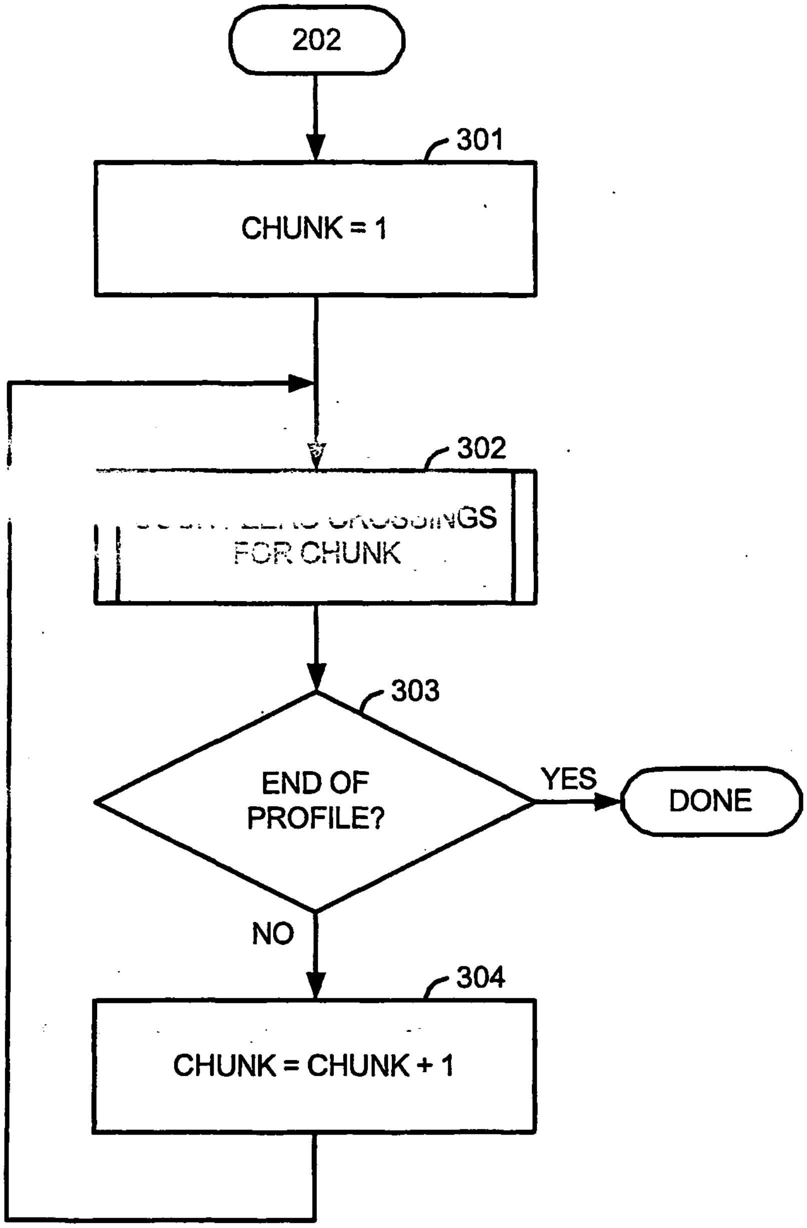
202
301
CHUNK = 1
302
COUNT ZERO CROSSINGS FOR CHUNK
303
END OF PROFILE?
YES
DONE
NO
304
CHUNK = CHUNK + 1

FIG.3

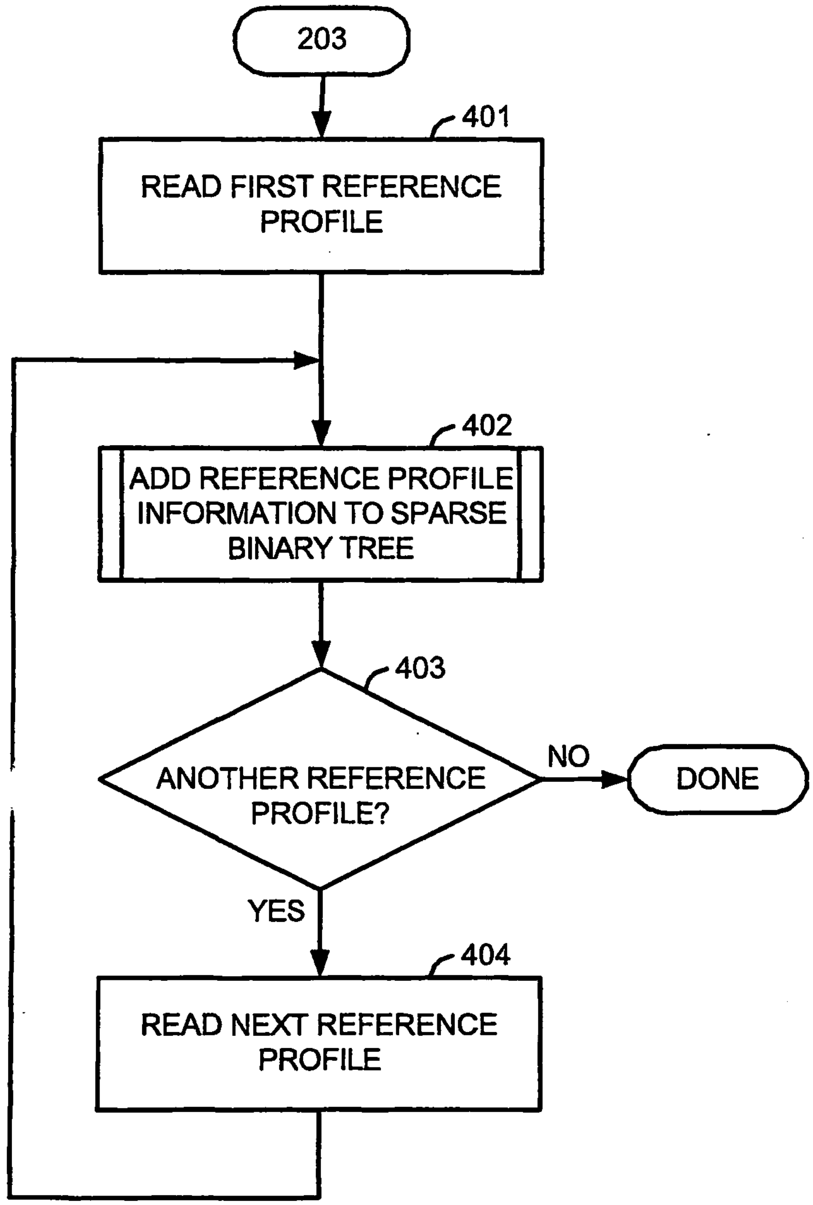
203
401
READ FIRST REFERENCE PROFILE
402
ADD REFERENCE PROFILE INFORMATION TO SPARSE BINARY TREE
403
ANOTHER REFERENCE PROFILE?
NO
DONE
YES
404
READ NEXT REFERENCE PROFILE

FIG.4

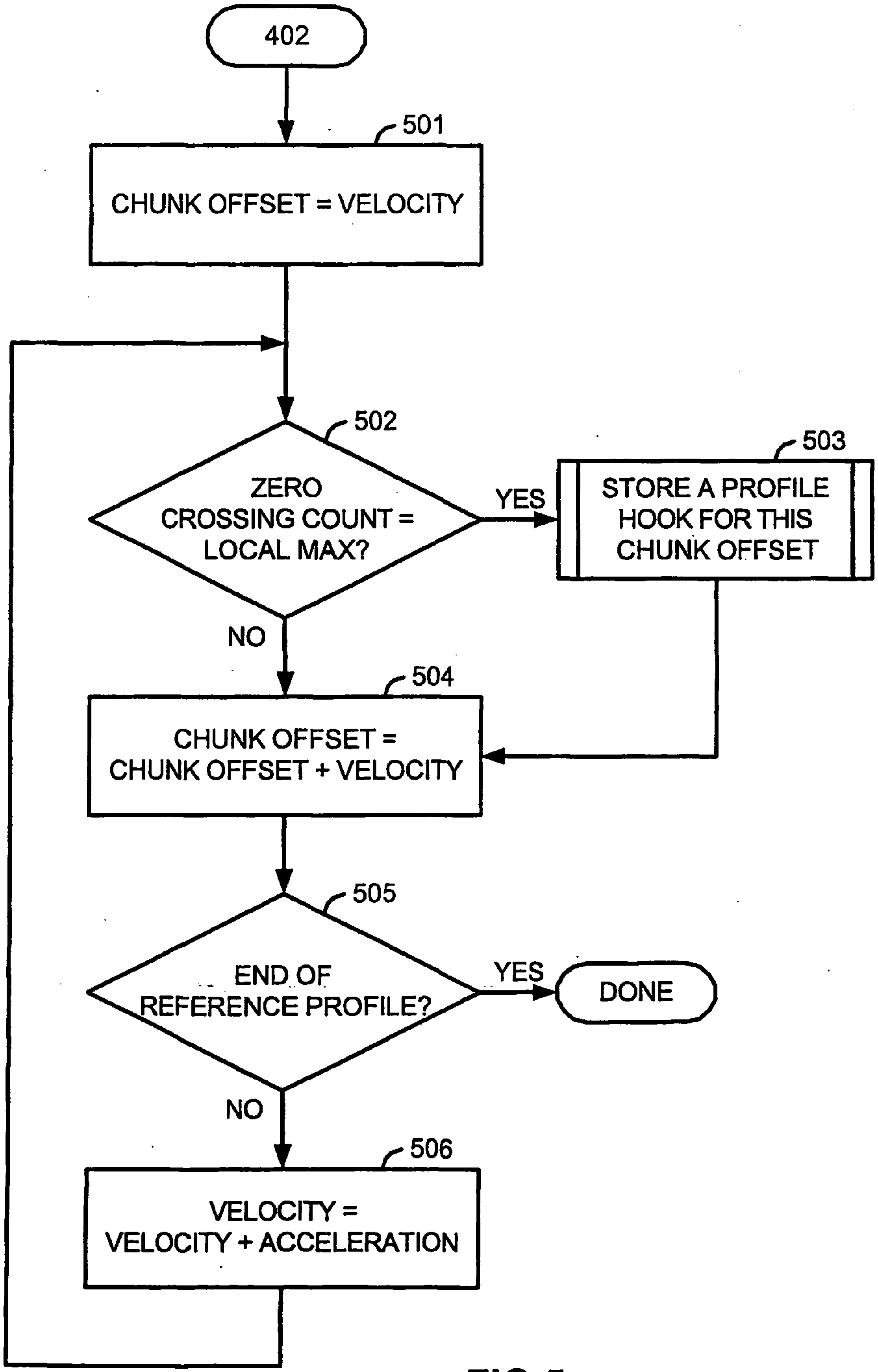
402
501
CHUNK OFFSET = VELOCITY
502
ZERO CROSSING COUNT = LOCAL MAX?
YES
503
STORE A PROFILE HOOK FOR THIS CHUNK OFFSET
NO
504
CHUNK OFFSET = CHUNK OFFSET + VELOCITY
505
END OF REFERENCE PROFILE?
YES
DONE
NO
506
VELOCITY = VELOCITY + ACCELERATION

FIG.5

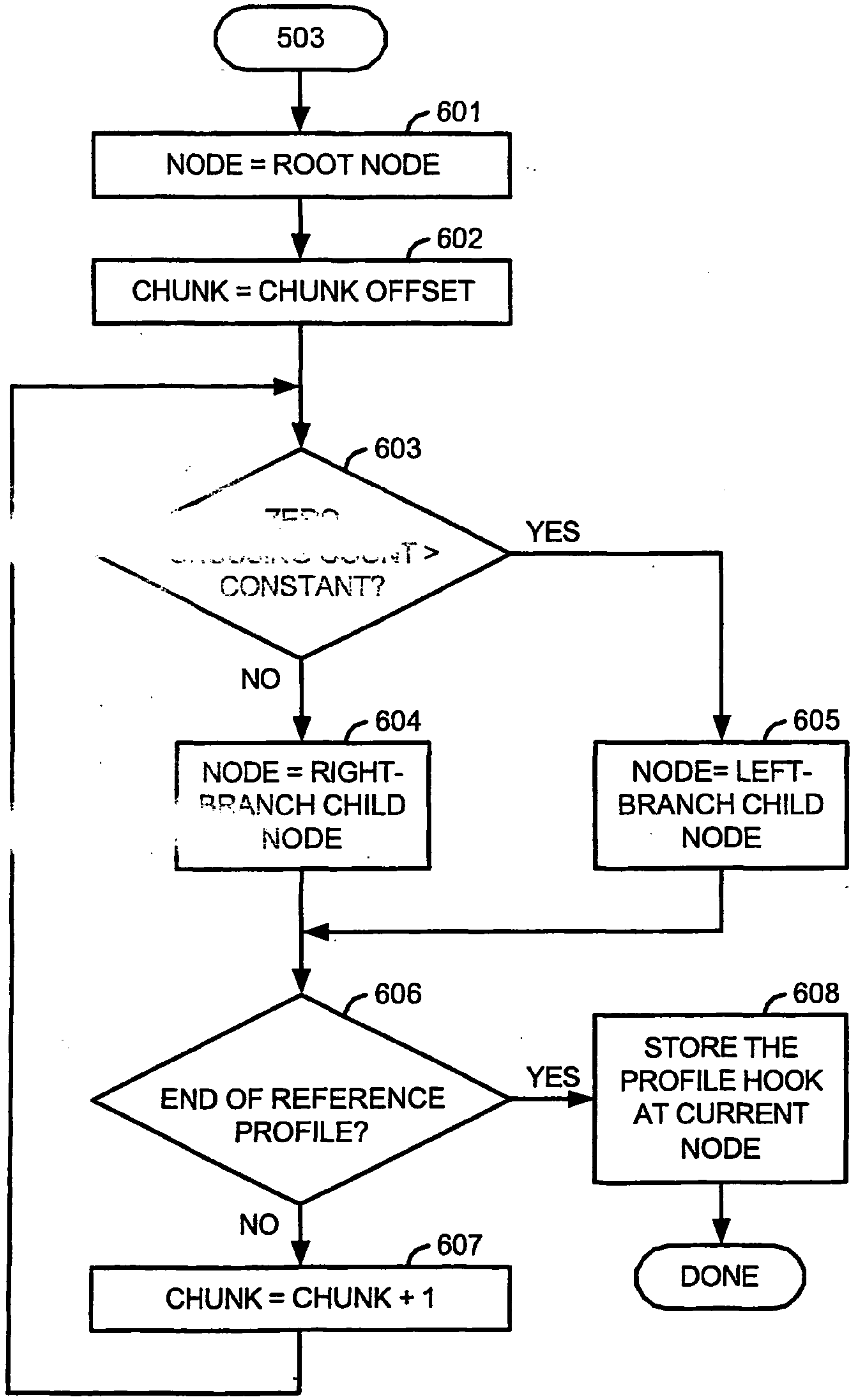
503
601
NODE = ROOT NODE
602
CHUNK = CHUNK OFFSET
603
ZERO CROSSING COUNT > CONSTANT?
YES
NO
604
NODE = RIGHT-BRANCH CHILD NODE
605
NODE= LEFT-BRANCH CHILD NODE
606
END OF REFERENCE PROFILE?
YES
NO
608
STORE THE PROFILE HOOK AT CURRENT NODE
607
CHUNK = CHUNK + 1
DONE

FIG.6

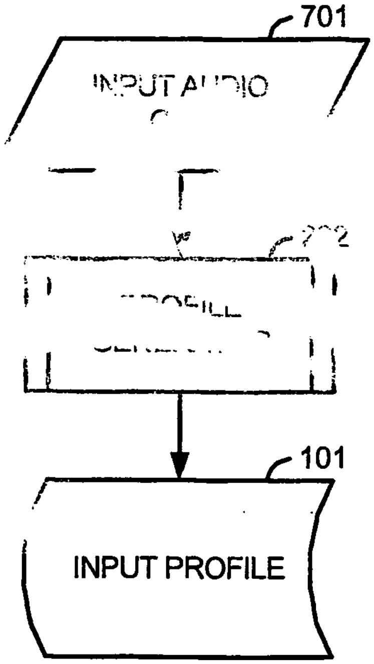

FIG.7

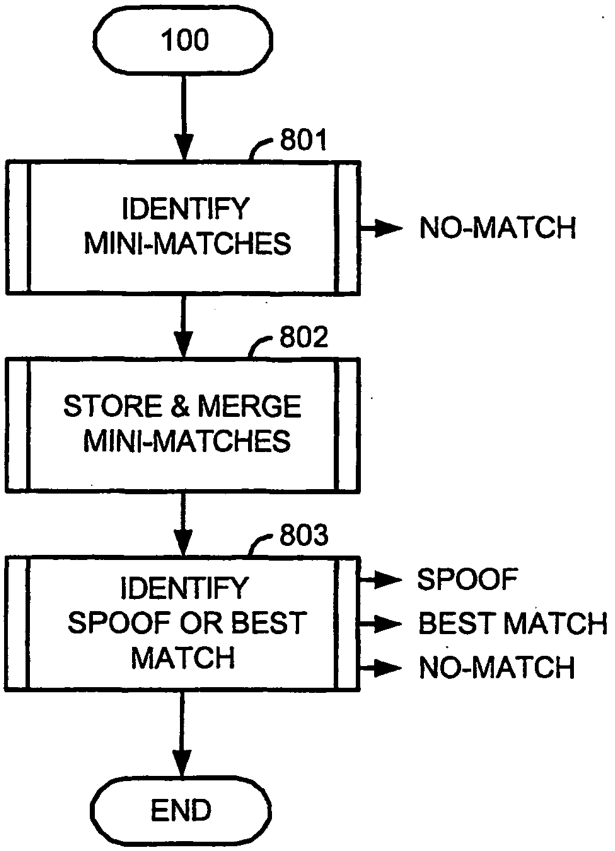
100
801
IDENTIFY MINI-MATCHES
NO-MATCH
802
STORE & MERGE MINI-MATCHES
803
IDENTIFY SPOOF OR BEST MATCH
SPOOF
BEST MATCH
NO-MATCH
END

FIG.8

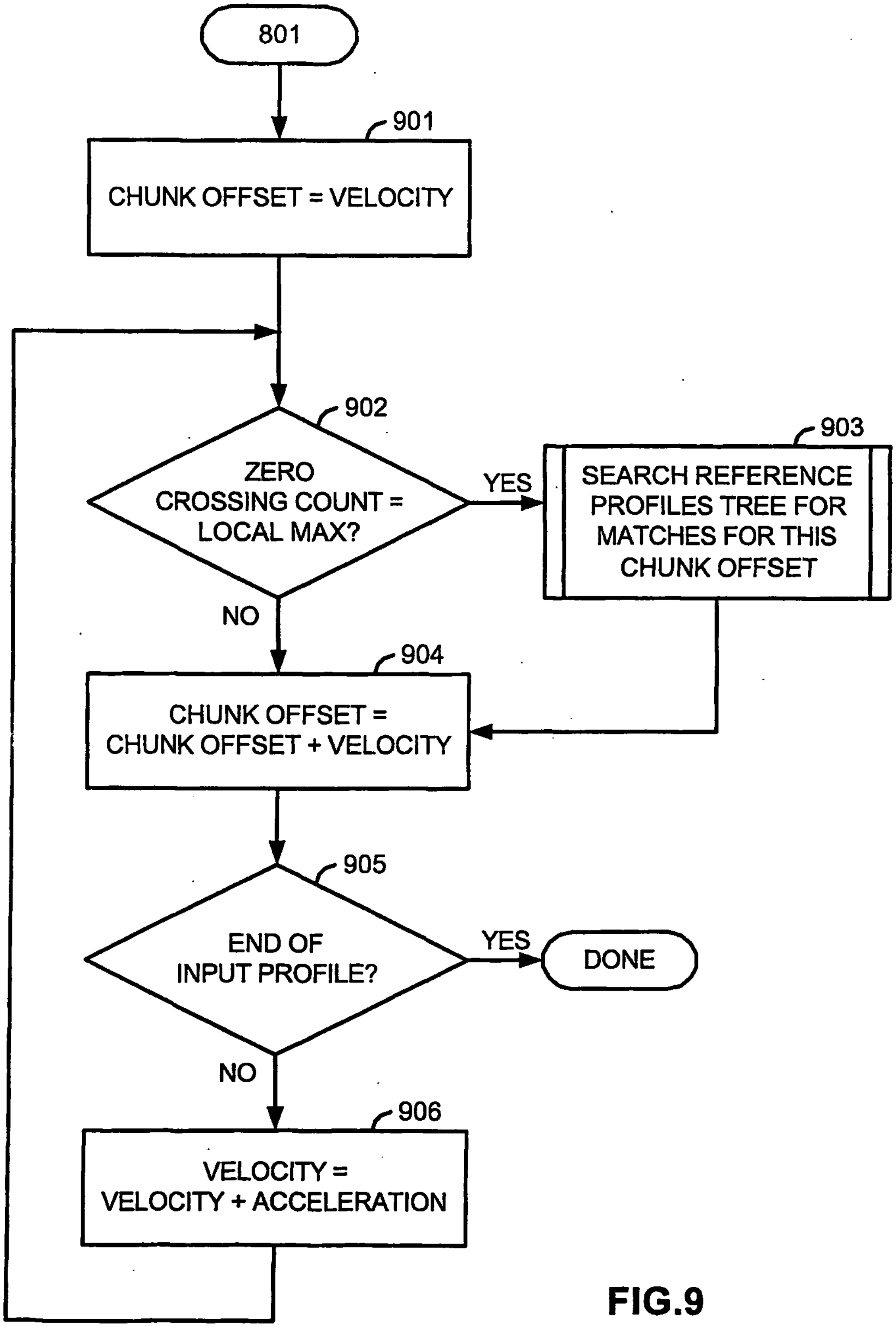
801
901
CHUNK OFFSET = VELOCITY
902
ZERO CROSSING COUNT = LOCAL MAX?
YES
903
SEARCH REFERENCE PROFILES TREE FOR MATCHES FOR THIS CHUNK OFFSET
NO
904
CHUNK OFFSET = CHUNK OFFSET + VELOCITY
905
END OF INPUT PROFILE?
YES
DONE
NO
906
VELOCITY = VELOCITY + ACCELERATION

FIG.9

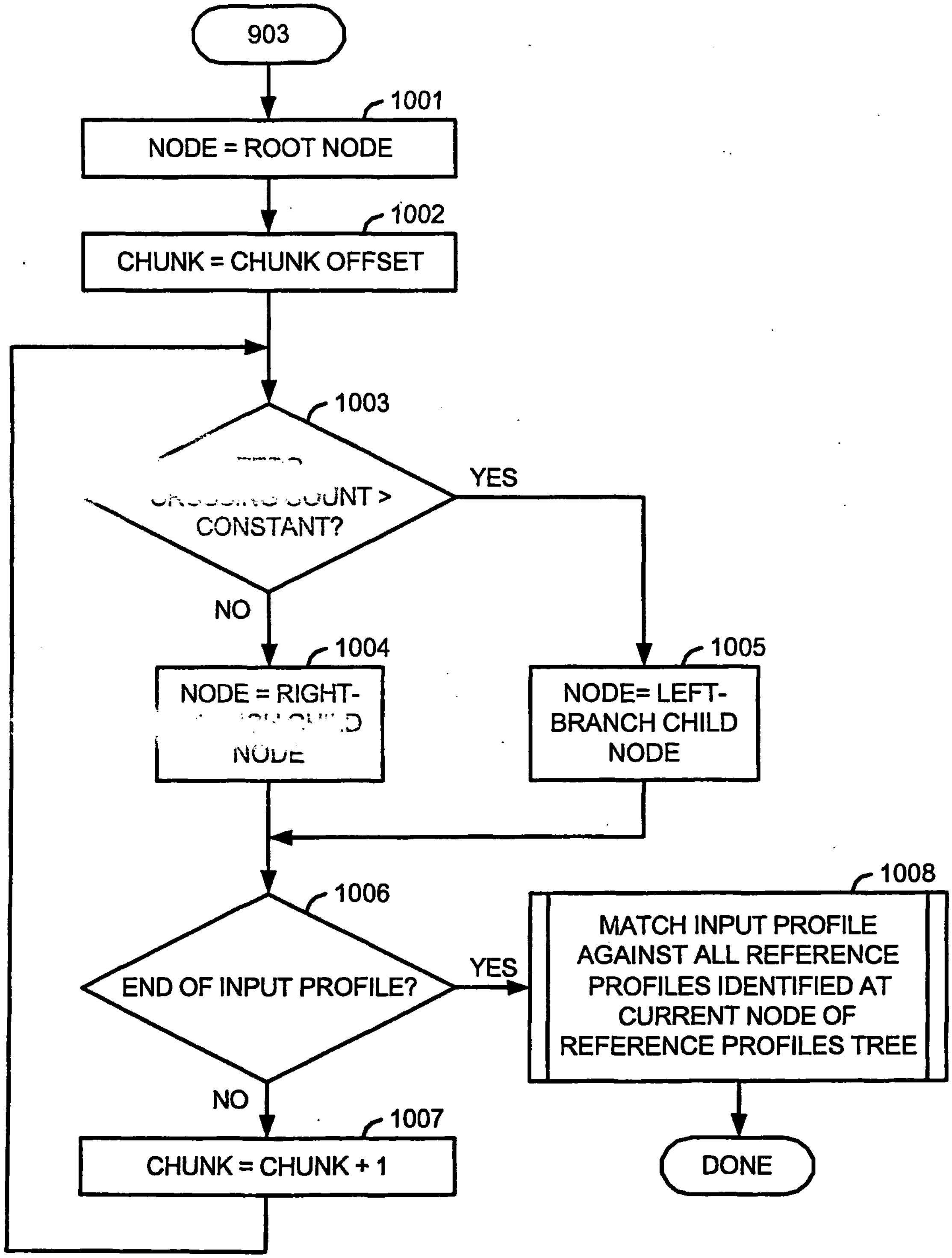
903
1001
NODE = ROOT NODE
1002
CHUNK = CHUNK OFFSET
1003
ZERO CROSSING COUNT > CONSTANT?
YES
NO
1004
NODE = RIGHT-BRANCH CHILD NODE
1005
NODE= LEFT-BRANCH CHILD NODE
1006
END OF INPUT PROFILE?
YES
1008
MATCH INPUT PROFILE AGAINST ALL REFERENCE PROFILES IDENTIFIED AT CURRENT NODE OF REFERENCE PROFILES TREE
NO
1007
CHUNK = CHUNK + 1
DONE

FIG.10

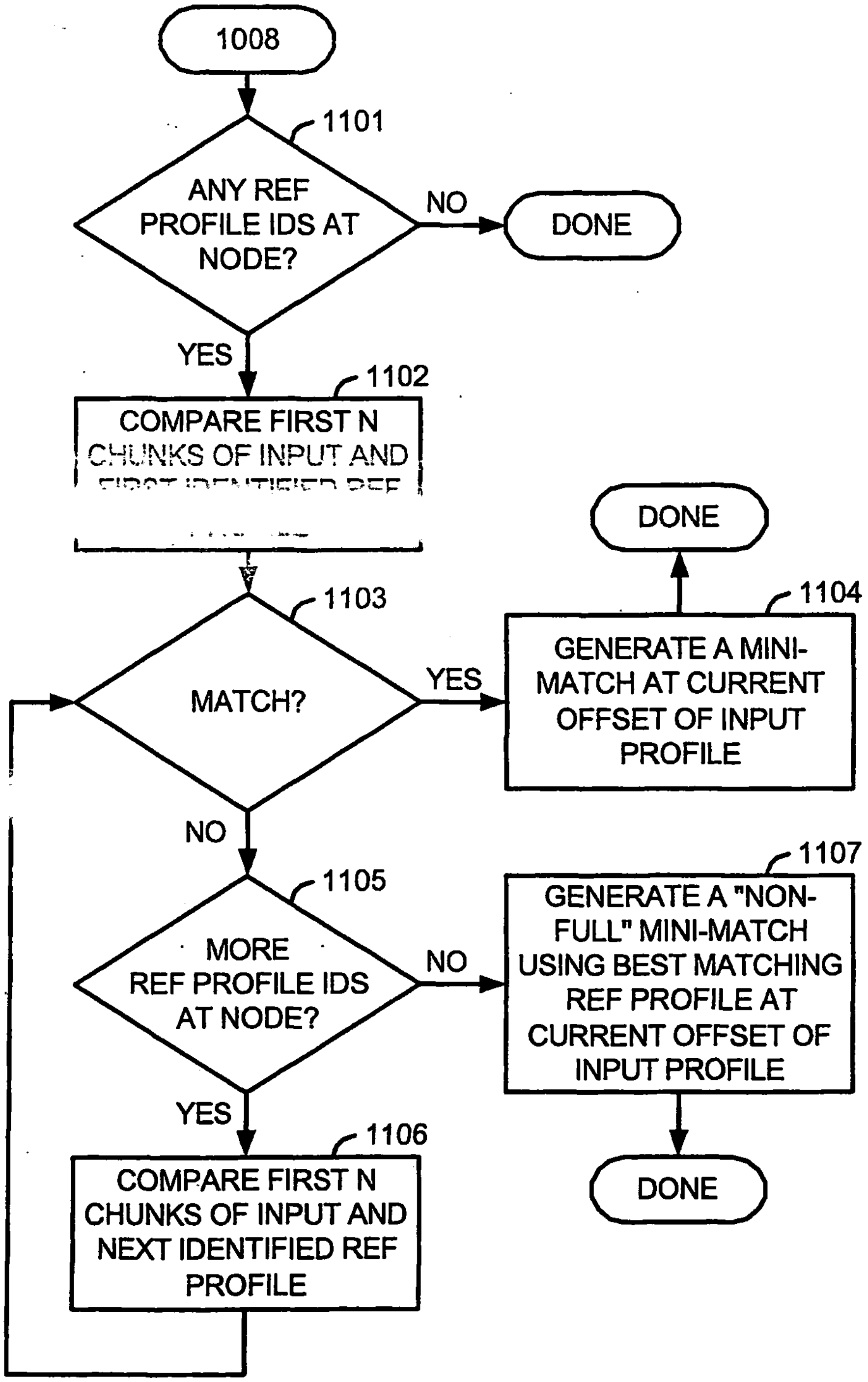
1008
1101
ANY REF PROFILE IDS AT NODE?
NO
DONE
YES
1102
COMPARE FIRST N CHUNKS OF INPUT AND FIRST IDENTIFIED REF
DONE
1103
MATCH?
YES
1104
GENERATE A MINI-MATCH AT CURRENT OFFSET OF INPUT PROFILE
NO
1105
MORE REF PROFILE IDS AT NODE?
NO
1107
GENERATE A "NON-FULL" MINI-MATCH USING BEST MATCHING REF PROFILE AT CURRENT OFFSET OF INPUT PROFILE
YES
1106
COMPARE FIRST N CHUNKS OF INPUT AND NEXT IDENTIFIED REF PROFILE
DONE

FIG.11

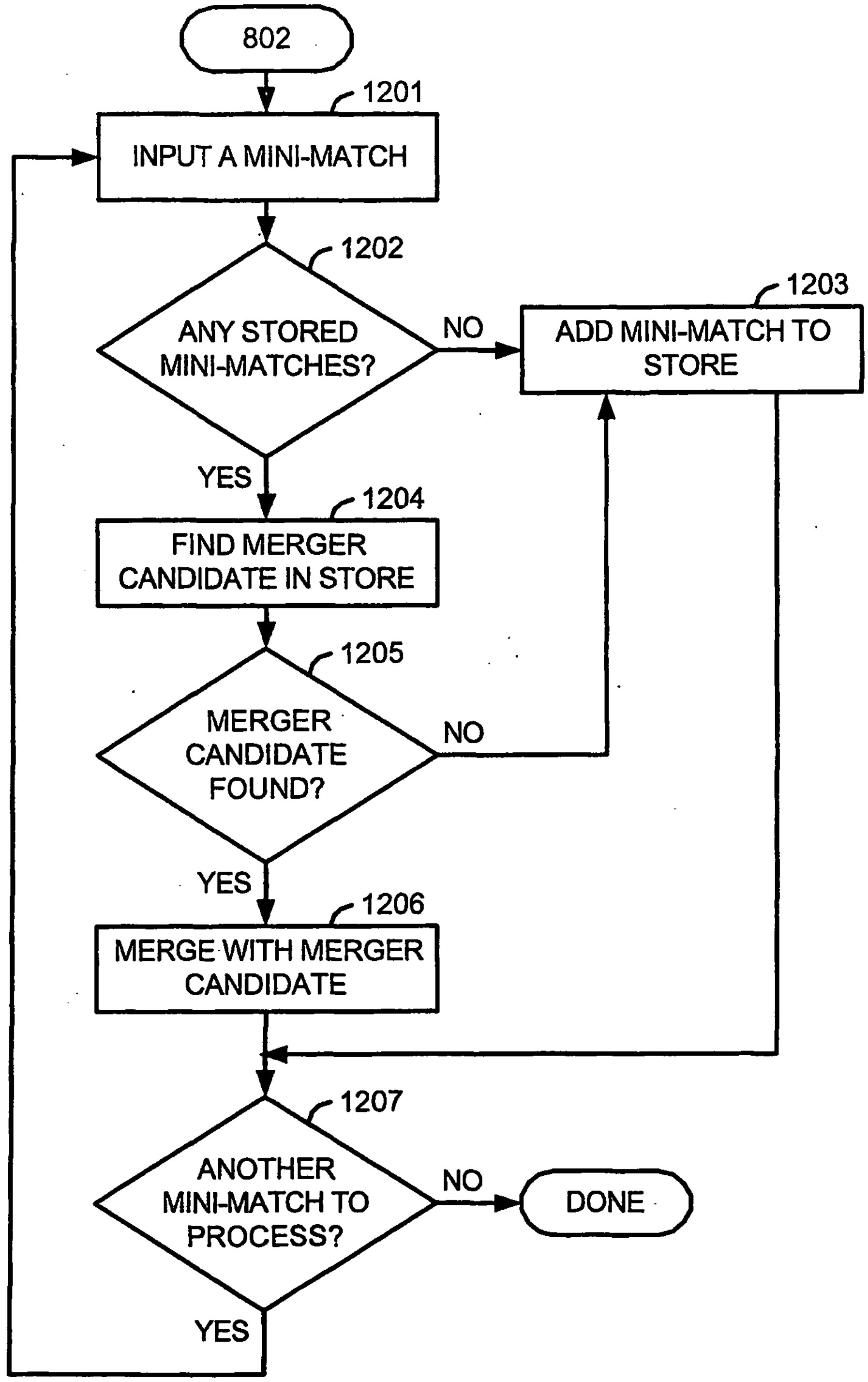
802
1201
INPUT A MINI-MATCH
1202
ANY STORED MINI-MATCHES?
NO
1203
ADD MINI-MATCH TO STORE
YES
1204
FIND MERGER CANDIDATE IN STORE
1205
MERGER CANDIDATE FOUND?
NO
YES
1206
MERGE WITH MERGER CANDIDATE
1207
ANOTHER MINI-MATCH TO PROCESS?
NO
DONE
YES

FIG.12

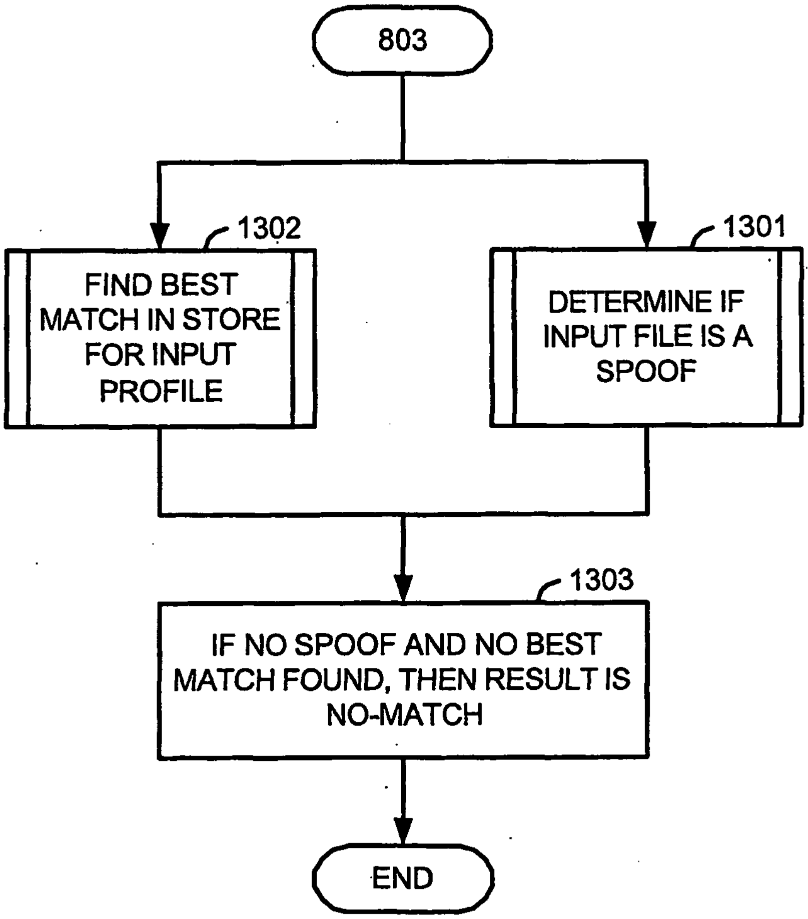
803
1302
FIND BEST MATCH IN STORE FOR INPUT PROFILE
1301
DETERMINE IF INPUT FILE IS A SPOOF
1303
IF NO SPOOF AND NO BEST MATCH FOUND, THEN RESULT IS NO-MATCH
END

FIG.13

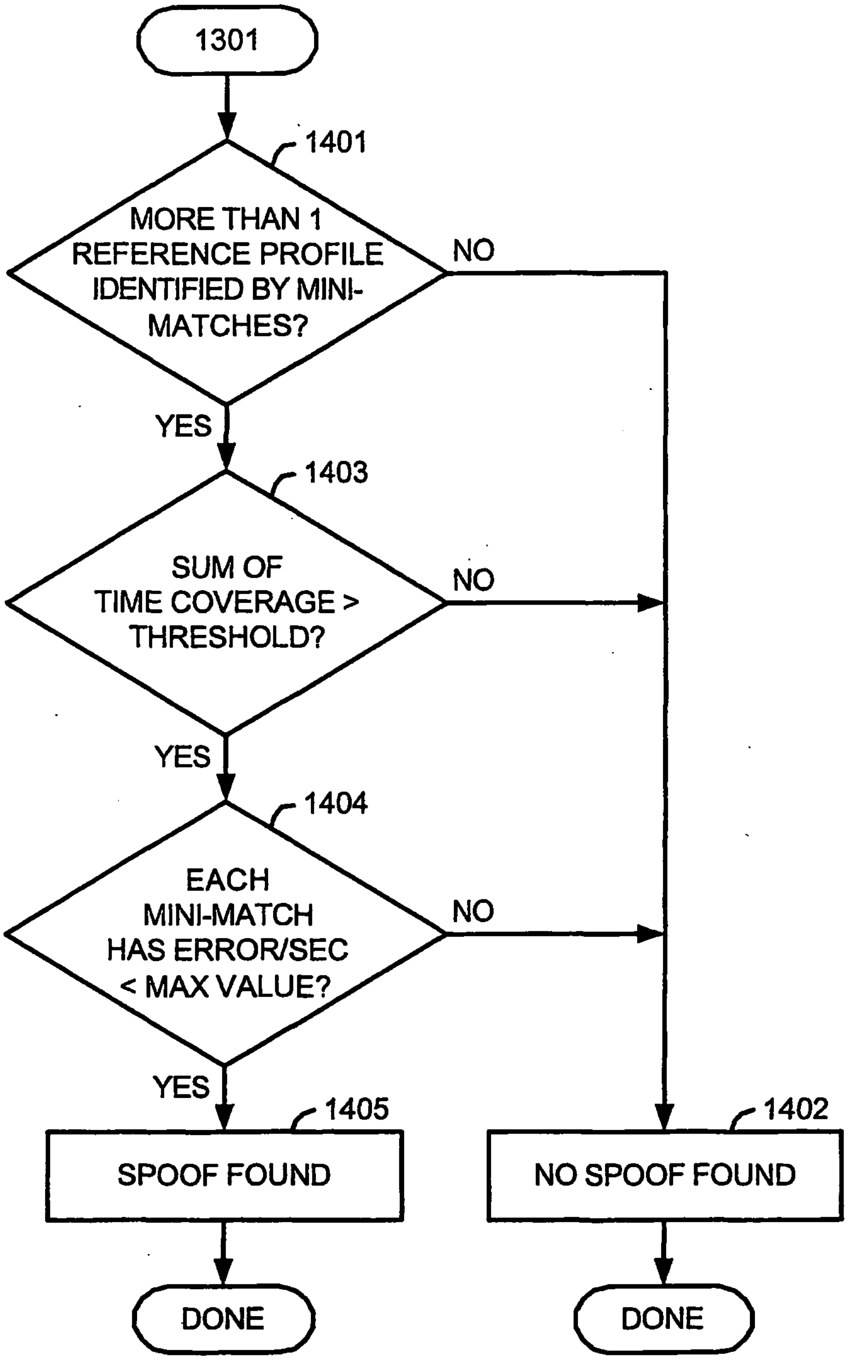
1301
1401
MORE THAN 1 REFERENCE PROFILE IDENTIFIED BY MINI-MATCHES?
NO
YES
1403
SUM OF TIME COVERAGE > THRESHOLD?
NO
YES
1404
EACH MINI-MATCH HAS ERROR/SEC < MAX VALUE?
NO
YES
1405
SPOOF FOUND
1402
NO SPOOF FOUND
DONE
DONE

FIG.14

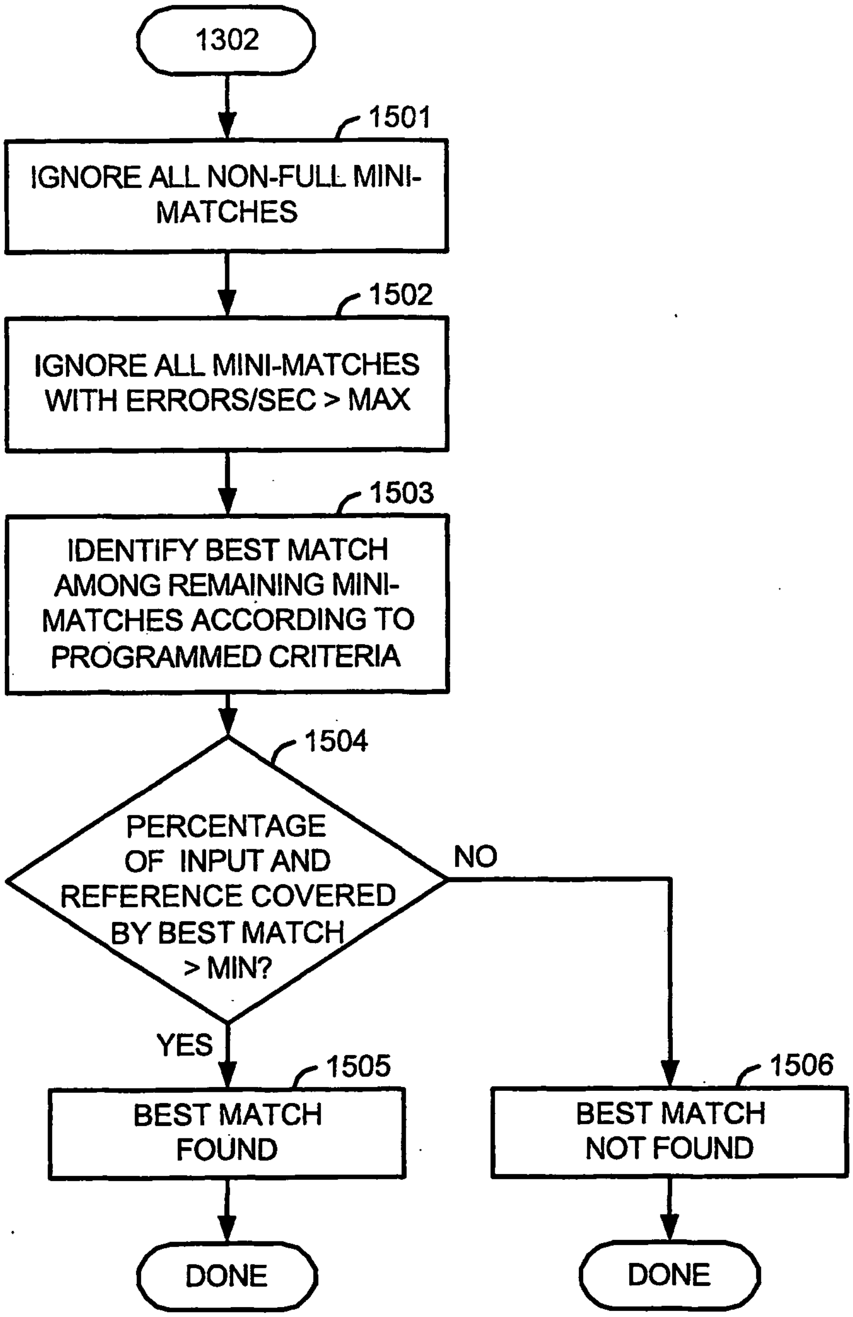
1302
1501
IGNORE ALL NON-FULL MINI-MATCHES
1502
IGNORE ALL MINI-MATCHES WITH ERRORS/SEC > MAX
1503
IDENTIFY BEST MATCH AMONG REMAINING MINI-MATCHES ACCORDING TO PROGRAMMED CRITERIA
1504
PERCENTAGE OF INPUT AND REFERENCE COVERED BY BEST MATCH > MIN?
NO
YES
1505
BEST MATCH FOUND
1506
BEST MATCH NOT FOUND
DONE
DONE

FIG.15